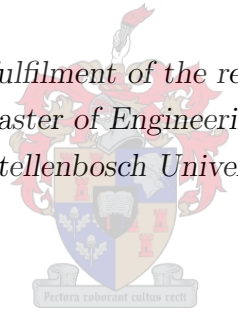
Occupancy Grid Mapping using Stereo Vision

by

Alwyn Johannes Burger

Thesis presented in partial fulfilment of the requirements for the degree of
Master of Engineering
at Stellenbosch University



Supervisors:

Dr C.E. van Daalen
Electrical and Electronic Engineering
Dr W.H. Brink
Mathematical Sciences

March 2015

Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the owner of the copyright thereof (unless to the extent explicitly otherwise stated) and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

March 2015

Acknowledgements

I would like to thank the following people and organisations, without whom this study would not have been possible:

- My supervisors for all their support and constructive criticism.
- The National Research Foundation of South Africa for their financial support.
- All of my colleagues at the Electronic Systems Laboratory for all the times not spent working.
- My family for their unconditional love and support.

Abstract

This thesis investigates the use of stereo vision sensors for dense autonomous mapping. It characterises and analyses the errors made during the stereo matching process so measurements can be correctly integrated into a 3D grid-based map. Maps are required for navigation and obstacle avoidance on autonomous vehicles in complex, unknown environments. The safety of the vehicle as well as the public depends on an accurate mapping of the environment of the vehicle, which can be problematic when inaccurate sensors such as stereo vision are used. Stereo vision sensors are relatively cheap and convenient, however, and a system that can create reliable maps using them would be beneficial.

A literature review suggests that occupancy grid mapping poses an appropriate solution, offering dense maps that can be extended with additional measurements incrementally. It forms a grid representation of the environment by dividing it into cells, and assigns a probability to each cell of being occupied. These probabilities are updated with measurements using a sensor model that relates measurements to occupancy probabilities.

Numerous forms of these sensor models exist, but none of them appear to be based on meaningful assumptions and sound statistical principles. Furthermore, they all seem to be limited by an assumption of unimodal, zero-mean Gaussian measurement noise.

Therefore, we derive a principled inverse sensor model (PRISM) based on physically meaningful assumptions. This model is capable of approximating any realistic measurement error distribution using a Gaussian mixture model (GMM). Training a GMM requires a characterisation of the measurement errors, which are related to the environment as well as which stereo matching technique is used. Therefore, a method for fitting a GMM to the error distribution of a sensor using measurements and ground truth is presented.

Since we may consider the derived principled inverse sensor model to be theoretically correct under its assumptions, we use it to evaluate the approximations made by other models from the literature that are designed for execution speed. We show that at close range these models generally offer good approximations that worsen with an increase in measurement distance.

We test our model by creating maps using synthetic and real world data. Comparing its results to those of sensor models from the literature suggests that our model calculates occupancy probabilities reliably. Since our model captures the limited measurement range of stereo vision, we conclude that more accurate sensors are required for mapping at greater distances.

Uittreksel

Hierdie tesis ondersoek die gebruik van stereovisie sensors vir digte outonome kartering. Dit karakteriseer en ontleed die foute wat gemaak word tydens die stereopassingsproses sodat metings korrek geïntegreer kan word in 'n 3D rooster-gebaseerde kaart. Sulke kaarte is nodig vir die navigasie en hindernisvermyding van outonome voertuie in komplekse en onbekende omgewings. Die veiligheid van die voertuig sowel as die publiek hang af van 'n akkurate kartering van die voertuig se omgewing, wat problematies kan wees wanneer onakkurate sensors soos stereovisie gebruik word. Hierdie sensors is egter relatief goedkoop en gerieflik, en daarom behoort 'n stelsel wat hulle dit gebruik om op 'n betroubare manier kaarte te skep baie voordelig te wees.

'n Literatuuroorsig dui daarop dat die besettingsroosteralgoritme 'n geskikte oplossing bied, aangesien dit digte kaarte skep wat met bykomende metings uitgebrei kan word. Hierdie algoritme skep 'n roostervoorstelling van die omgewing en ken 'n waarskynlikheid dat dit beset is aan elke sel in die voorstelling toe. Hierdie waarskynlikhede word deur nuwe metings opgedateer deur gebruik te maak van 'n sensormodel wat beskryf hoe metings verband hou met besettingswaarskynlikhede.

Menigde afleidings bestaan vir hierdie sensormodelle, maar dit blyk dat geen van die modelle gebaseer is op betekenisvolle aannames en statistiese beginsels nie. Verder lyk dit asof elkeen beperk word deur 'n aanname van enkelmodale, nul-gemiddelde Gaussiese metingsgeraas.

Ons lei 'n beginselfundeerde omgekeerde sensormodel af wat gebaseer is op fisies betekenisvolle aannames. Hierdie model is in staat om enige realistiese foutverspreiding te weerspieël deur die gebruik van 'n Gaussiese mengselmodel (GMM). Dit vereis 'n karakterisering van 'n stereovisie sensor se metingsfoute, wat afhang van die omgewing sowel as watter stereopassingstegniek gebruik is. Daarom stel ons 'n metode voor wat die foutverspreiding van die sensor met behulp van 'n GMM modelleer deur gebruik te maak van metings en absolute verwysings.

Die afgeleide ge inverteerde sensormodel is teoreties korrek en kan gevolglik gebruik word om modelle uit die literatuur wat vir uitvoerspoed ontwerp is te evalueer. Ons wys dat op kort afstande die modelle oor die algemeen goeie benaderings bied wat versleg soos die metingsafstand toeneem.

Ons toets ons nuwe model deur kaarte te skep met gesimuleerde data, sintetiese data, en werklike data. Vergelykings tussen hierdie resultate en dié van sensormodelle uit die literatuur dui daarop dat ons model besettingswaarskynlikhede betroubaar bereken. Aangesien ons model die beperkte metingsafstand van stereovisie vasvang, lei ons af dat meer akkurate sensors benodig word vir kartering oor groter afstande.

Nomenclature

Abbreviations

AIC	Akaike information criterion
EM	expectation maximisation
GMM	Gaussian mixture model
ISM	inverse sensor model
LIDAR	light detection and ranging
OR	occupancy ratio
PRISM	principled inverse sensor model
SLAM	simultaneous localisation and mapping

Notations

x	scalar
$\underline{\mathrm{x}}$	vector
$\widetilde{\underline{\mathrm{x}}}$	Euclidean version of homogeneous vector $\underline{\mathrm{x}}$
$\underline{0}$	zero column vector
X	matrix
K	calibration matrix
R	rotation matrix
P	projection or camera matrix
X	axis
X_w, Y_w, Z_w	world axes
X_c, Y_c, Z_c	camera axes
$X_{\mathrm{im}}, Y_{\mathrm{im}}$	image axes
X_n, Y_n, Z_n	rectified camera axes

$\underline{\mathrm{x}}_w = \begin{bmatrix} x_w & y_w & z_w \end{bmatrix}^T$	point in world coordinates
$\underline{\mathrm{x}}_c = \begin{bmatrix} x_c & y_c & z_c \end{bmatrix}^T$	point in camera coordinates
$\underline{\mathrm{x}}_{\mathrm{im}} = \begin{bmatrix} x_{\mathrm{im}} & y_{\mathrm{im}} \end{bmatrix}^T$	point in image coordinates
$\underline{\mathrm{x}}_n = \begin{bmatrix} x_n & y_n & z_n \end{bmatrix}^T$	point in rectified camera coordinates
$\underline{\mathrm{x}}_{\mathrm{L}} = \begin{bmatrix} x_{\mathrm{L}} & y_{\mathrm{L}} \end{bmatrix}^T$	point in image coordinates in left image
$\underline{\mathrm{x}}_{\mathrm{R}} = \begin{bmatrix} x_{\mathrm{R}} & y_{\mathrm{R}} \end{bmatrix}^T$	point in image coordinates in right image
$\underline{\mathrm{p}} = \begin{bmatrix} p_x & p_y \end{bmatrix}^T$	principal point
b	baseline
f	focal length
$\mathbf{m}_i$	binary random variable describing the occupancy of cell i
$z_{1:t}$	set of measurements from time step 1 to time step t

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Autonomous robots offer many applications for humanity by performing tasks deemed unsafe or impossible for humans. For example, robots are used autonomously in mines [1,2], space exploration [3,4], and in medical applications [5, 6]. These robots are capable of functioning without human intervention or assistance, which offers exciting new opportunities. If their autonomous functions get more sophisticated, it will be safe to integrate them more into society, and even have them interact with the public. This will improve the commercialisation of these robots.

For a robot to be considered truly autonomous, it should be capable of sensing and perception. This is required for localising the robot within its environment, and to allow route planning and obstacle avoidance. Much research has been done to accomplish this using a variety of sensors – from close range sonar [7–9] to highly accurate scanning lasers [10, 11].

Because of the accuracy of its measurements, light detecting and ranging (LIDAR) is a popular sensor for mapping the environments of robots [12]. These sensors are relatively expensive, however, and their power usage and size make them impractical for smaller robots.

Cameras are a practical sensor for robot perception due to their relatively small size and low price. Computer vision can use either a single camera or an array of cameras to allow a robot to 'see' the world around it and to estimate the distances to visible objects.

In robotics, perceiving the environment with cameras is often limited to a few points that can be detected and tracked – instead of perceiving the surroundings of the robot in full. It would seem, however, that complex robots need dense maps of their environments to navigate precisely and safely. For instance, robots capable of climbing stairs or using tunnels have been shown to require dense 3D maps [13].

Since robots are increasingly being integrated with the world, it is becoming essential for them to be capable of navigating without endangering people. This, and the increasing complexity of the tasks that they perform, makes an accurate representation of their environment more important than ever.

There is currently no accurate way to create dense maps of a robot's environment using cameras. The measurements from cameras are noisy, which makes the true locations of measured objects uncertain. Therefore, the errors in measurement made by the cameras must be considered carefully when creating maps, which makes the creation of accurate maps much more complex.

1.2 Aims

We aim to investigate the autonomous creation of maps for mobile robots using computer vision. To improve the mapping accuracy, any uncertainty in measurements must be investigated and incorporated into the mapping process. A system capable of reliably representing the surroundings of a robot would be greatly beneficial for the field, particularly if dense maps are created.

Stereo vision offers an interesting alternative sensor for creating a dense environment map, partly due to the large number of range measurements it produces per time step. It consists of two cameras whose images are compared to find the distances to nearby objects. These cameras are often already present on autonomous vehicles due to their application to localisation [14–16], where they are used to track the locations of a number of features to estimate the pose of the vehicle. In these cases this would mean that implementing a mapping system with cameras would require no additional hardware, which decreases implementation cost and weight.

The main goal of this study is to develop a system that uses stereo vision to create a dense map that can be used in route planning and collision avoidance. Since dense stereo vision data is known for being inaccurate at further distances, specific emphasis will be placed on incorporating the uncertainty in measurements into the calculated map. In order to accomplish this, the errors made by the sensor will be studied so measurements may be used as effectively as possible.

We also aim to evaluate how useful stereo vision can be as a sensor for dense mapping, when compared to existing systems that use LIDAR. If its errors are too large, stereo vision may not be considered as a good sensor to use for autonomous mapping. Therefore, special attention will be paid to the distribution of errors, as well as the effect they have on created maps.

1.3 System Overview

The system that we use to create maps from stereo vision is shown in Figure 1.1. We briefly explain here the objectives we set, and provide an overview of the method we use. The motivations for the design decisions mentioned here are given later in the thesis.

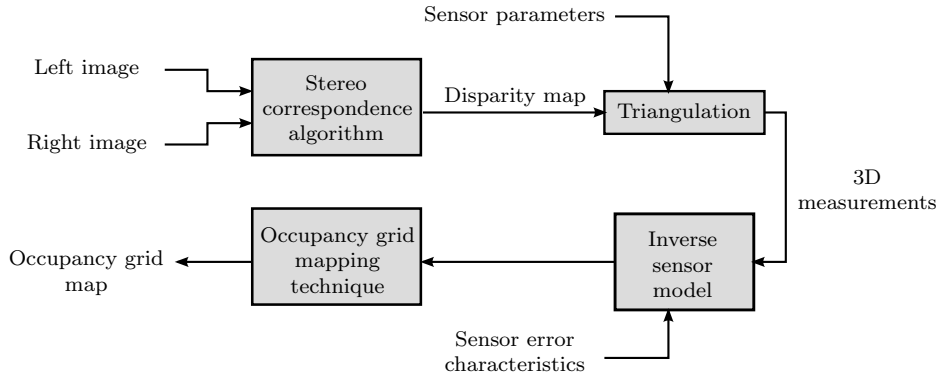


Figure 1.1: System framework

The first step is to capture an image pair from two cameras, and an example of such an image pair from the KITTI dataset [12] is shown in Figure 1.2. The basic principle of stereo vision is that two images are taken from different viewpoints. The relative horizontal position of a point in the two images, called the pixel disparity, can then be used to estimate the point's distance from the sensor.

Figure 1.2: Example of a stereo image pair that can be used to estimate the distances of visible objects.

The process of finding such points in the two images is called the stereo correspondence problem, and is the focus of much research. The purpose of these techniques is to find the image correspondences as quickly as possible, without sacrificing the accuracy of the results. Measured disparities will inevitably contain errors, though, due to how difficult is it to find all corresponding points exactly.

The output of the stereo correspondence algorithm is a disparity map similar to the one shown in Figure 1.3, where a darker colour indicates a closer point and white points are not calculated. This disparity map collects all the measured disparity values calculated for an image pair. Using the disparity of a pixel, a point can be triangulated to real world coordinates. This process results in a cloud of 3D points that are measurements of objects in the environment.

Figure 1.3: Example of a disparity map.

If these measurements were perfect, these points could be used directly, but they are known to be imperfect. Therefore, a characterisation of the sensor – called an inverse sensor model (ISM) – is used that relates this noisy measurement to probabilities that an object is located at different positions.

An example of such an ISM is shown in Figure 1.4, displaying the likelihood that an object is located at different distances for a measured distance of 1 metre. It shows that since no objects were present in the range between the sensor and the measurement, this range is probably empty. Near the measurement the chances that an object exists are considerably higher, and for ranges further away from the sensor a value of 0.5 indicates that no information about the presence of objects is available. Using these probabilities, the inverse sensor model provides the information we use to update the map.

The occupancy grid mapping technique is a popular way to represent an environment [17–19]. It divides the environment into cells and assigns a probability to each cell that an object exists within it. These probabilities are updated over different measurements by using the probabilities calculated with the ISM. Visualising the probabilities of the map cells yields Figure 1.5, which shows the occupancy grid

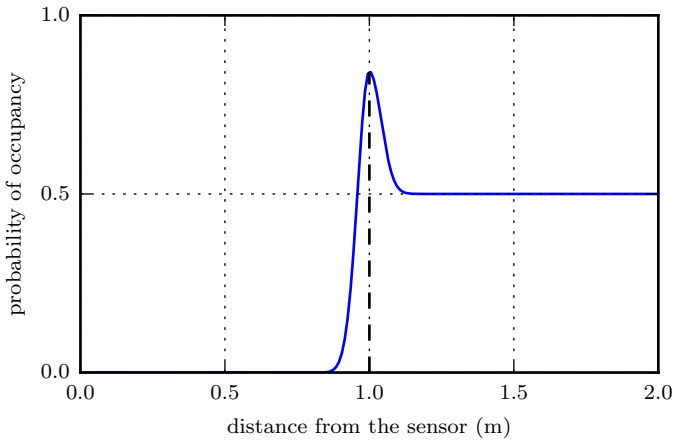


Figure 1.4: Example of an inverse sensor model.

for a different example. In this visualisation, white indicates that a cell is empty, black indicates a cell is occupied by an object, and grey means that uncertainty exists about the occupancy of that cell.

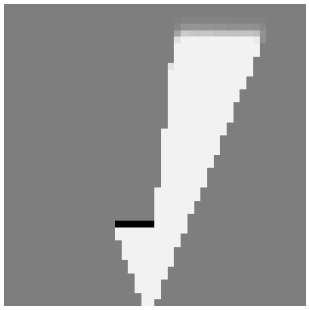


Figure 1.5: Example of an occupancy grid map.

1.4 Thesis Overview

Firstly, we investigate the most prominent existing techniques and projects from the literature in Chapter 2. We make some design choices regarding which techniques to use for this study, and conclude with a detailed problem statement that specifies what this study hopes to accomplish.

In Chapter 3 we discuss dense stereo as a sensor for mapping, and analyse the measurement errors that some correspondence algorithms make using two different datasets. We expect that an accurate probabilistic characterisation of these errors should improve mapping.

We discuss mapping using an occupancy grid map discussed in Chapter 4, with specific focus on how new measurements are incorporated into the map. We also specify and explain the assumptions that we make.

After this, we investigate the modelling of the stereo vision measurements in detail in Chapter 5. When calculating the map, we need to accurately reflect the measurements errors, or else the result will not be reliable.

Results of testing the mapping system are shown in Chapter 6, where we analyse the maps. Lastly, we draw some conclusions about the study and suggest possibilities for further work in Chapter 7.

Chapter 2

Literature Study

Mapping on autonomous robotics with cameras has attracted great interest recently, partly due to advancements in computing power and optics. These developments have led to more advanced systems capable of previously impossible complexity. Some prominent and popular techniques from the literature are discussed in this chapter to offer an overview of the relevant fields. Section 2.1 gives a summary of some of the most prominent algorithms in stereo vision, Section 2.2 discusses various mapping techniques, and Section 2.3 highlights sensor modelling. After this, we investigate some related projects in Section 2.4, and provide a detailed problem statement in Section 2.5.

2.1 Stereo Vision

A method of measuring the distances that objects are from the sensor is an essential part of any autonomous mapping system. Stereo vision is a popular choice, since it offers large amounts of information per time step when compared to other methods such as sonar or radar, and is generally less expensive than LIDAR.

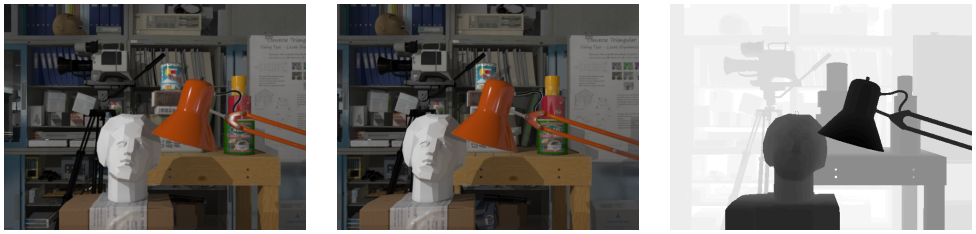
After discussing some core concepts required for stereo vision, we look at some common algorithms from the literature.

2.1.1 Core Concepts

The basic idea behind stereo vision is to use two cameras to estimate how far objects are. This is done by using the parallax effect, where objects viewed simultaneously from two parallel viewpoints appear in different positions [20]. Closer objects have a wider discrepancy in apparent horizontal positions.

Example images from two such cameras are shown in Figure 2.1. Any technique that uses an object's disparity in two views can be labelled as stereo vision, but dense stereo is specifically when it attempts to calculate a disparity value for each pixel. The objective of a dense stereo algorithm is to construct a disparity map, which is a grid of disparity values with the same shape as the source image. An example of such a disparity map is shown in Figure 2.1(c).

One of the stereo images is usually considered as the reference image and the other as the target image. Finding disparity values is commonly referred to as the correspondence problem, since the aim is to find the corresponding positions in the target image for pixels in the reference image.



(a) (b) (c)

Figure 2.1: Sample stereo images from the Tsukuba dataset [21]. The left and right stereo images are shown in (a) and (b), with the disparity map in (c) indicating the distance to each element as a luminance value where lighter indicates further objects.

2.1.2 Common Algorithms

Calculating disparity values is not a trivial task and many different algorithms have been developed. It is complicated by a number of practical issues such as homogeneous areas in the environment, camera noise and bad lighting. It can also be difficult to find a universal method for quantifying and comparing similarities between image regions. For this study, some representative techniques that are suitable for autonomous mapping will be chosen from the literature.

A distinction can be made between localised and global methods [22, 23]. Localised methods find matching pixels by considering small areas around them in each image. Global methods consider the entire image and try to find matches for all the pixels simultaneously, usually by using energy minimisation. This involves setting up a function that penalises certain unwanted aspects of matches, and then finding an optimal solution for it.

Another distinction is between sparse and complete techniques, where an algorithm that finds disparities more sparsely results in less data. These results tend to be more accurate, though, since any pixel for which a clear match cannot be found is marked as incalculable.

Algorithms also make a trade-off between accuracy and execution speed. For our application the stereo matching will need to be fast to allow for online calculation, but if the values are not sufficiently accurate it may not be suitable to mapping.

2.1.2.1 Block Matching

The block matching algorithm of Konolige [24] is based on the classic correlation-based technique that has been used extensively [25, 26]. It is a local method that considers areas around the considered pixels in the two images to find the best matches.

Pixel comparisons are done by calculating the correlation between surrounding areas. The point of maximum correlation then relates to the part of the target image that most resembles the area around the reference pixel. Subpixel accuracy can be achieved in block matching by fitting a quadratic curve to a few best matches and solving for the local maximum.

The size of the area considered has a significant impact on the result, since a smaller area will be more likely to match and a larger area will result in a smoother result. Therefore the block matching technique requires some tweaking for a given application to perform optimally.

2.1.2.2 Semiglobal Block Matching

Semiglobal block matching, as developed by Hirschmüller [27], extends upon the block matching technique by also considering the disparity values of surrounding pixels. This leads to smoother, less noisy disparity maps, assuming that the distances to visible objects vary mostly smoothly.

The extension is done by minimising an energy function that considers not only the area around pixels as before, but also the local changes in disparity that a possible match would cause [28]. The function is calculated as the sum of the energy costs for a number of paths from different directions to the pixel in question, and for a defined range of possible disparity values. This leads to a range of energy costs at each pixel – one per possible disparity value – and the minimum cost found represents the best match.

Semiglobal block matching requires a more complex calculation process than traditional block matching, which leads to a longer execution time. Since it is not fully global, however, it still offers direct solving as opposed to an iterative solution that would require a series of approximating results.

2.1.2.3 Dynamic Programming

The first application of dynamic programming to stereo vision was done by Ohta and Kanade [29], matching stereo images by comparing pixels directly and considering the edges between different visible objects. This process involves repeatedly subdividing the image into smaller pieces and matching each piece separately before combining the results, which significantly reduces the complexity of the solution.

The problem with this simplification is that it assumes independence between the lines in the image. Each line in the reference image is matched individually, which means that the calculated disparity map is not necessarily smooth.

2.1.2.4 Belief Propagation

A more modern development in stereo vision is to make use of Markov random fields to model the similarities between the left and right images. Each random variable in the field represents the disparity of a particular pixel. Edges are defined between adjacent pixels and describe the relationships between them, to define local smoothness. One of the techniques for finding the lowest global matching cost in such a Markov random fields is belief propagation, as first described by Sun et al. [30].

The problem with belief propagation is that it requires iteration to converge to a solution, which makes it inconvenient for online calculation. Yang et al. developed it further by using a constant space algorithm [31], where the same memory is reused to limit the amount of data being stored. This led to significant reductions in the execution time and memory usage, but also decreased the accuracy of results due to further approximations.

2.1.2.5 Variational Matching

Variational matching also balances smoothness with locally accurate matching, combining costs for the differences between local areas around pixels with a smoothness constraint [32]. It compares regions around pixels by calculating the sum of squared differences. Smoothness is ensured by employing a regulariser that uses the gradient of the disparity map to ensure minimal local disparity variations. This smoothness value is then added to the data cost to create the energy function, which is solved by defining the disparity map as an Euler-Lagrange equation [33] and solving the resulting partial differential equation.

It is a local method, offering very fast calculation times and low memory usage. Results are fairly accurate [32] with some fuzziness at depth borders.

2.1.2.6 Conclusion

For the purposes of this study a few techniques are chosen for further investigation, representing both local and global, dense and sparse, as well as accurate and inaccurate techniques. We choose block matching, semiglobal block matching and variational matching. Although other techniques such as belief propagation offer very accurate results, they rely on iteratively solving large scale optimisation problems and do not offer practically feasible solutions for online mapping. Furthermore, implementations for the chosen techniques are available in the open source OpenCV library [34].

2.2 Mapping

An autonomous vehicle that navigates and investigates its surroundings autonomously needs a way to represent its environment given measurements. This is commonly done using a map that is capable of describing a region. Preferably, this should be in a memory efficient way which would allow for larger areas to be explored in more detail.

Some commonly used mapping techniques are discussed here to find the best option based on the requirements of this application. Each mapping technique will be evaluated based on its ability to store large amounts of data efficiently and accurately, the accessibility of its information, and how flexible the stored map is. To be considered flexible, it must be extendible when additional areas are explored, and its resolution should be variable so it can fit many different applications.

2.2.1 Point Clouds

The naive way to represent data points in world coordinates is to use a point cloud, where every known point in the map is represented with a 3D coordinate. Although it offers a very simple way to represent information and does not require additional processing, it provides limited usability for navigation and processing techniques due to the complexity of accessing information about a particular region [35].

The problem with point clouds is that they do not combine measurements that represent essentially the same information, and therefore its memory usage can grow without bound. Eventually, many points will contain redundant information.

Pan et al. [36] showed that it is computationally expensive to do collision detection with point clouds, since each point has to be considered individually. In order to know if an obstacle exists at a specific point, or to find the closest data point to a coordinate, every existing data point needs to be queried (leading to a complexity of $\mathcal{O}(n^2)$). Some simple improvements, such as combining points that lie close together or labelling data based on location using KD-trees, can be made to simplify usage. However, sophisticated processes like path planning are unnecessarily complicated when the environment is represented with individual points.

2.2.2 Topological Maps

One way to reduce the amount of data stored is to create a topological map [37, 38] that represents the environment with a graph. It works similarly to a roadmap, creating nodes of known or important points and connecting them with edges that represent pathways. This simplification makes topological maps

difficult to use, and some data is lost. For instance, they do not indicate the locations of objects and hence cannot be used for collision avoidance.

The graph that describes the map is very efficient to traverse, since each node is only connected to a few others. This makes them convenient to use for navigation algorithms, where paths through the graph are often used to represent possible routes. Applications of this also includes transport networks like rail systems or subways.

Another issue with topological mapping is its inability to represent uncertainty. Although it represents known positions well, it is hard to distinguish between empty space and unknown areas.

2.2.3 Occupancy Grids

Currently the most popular solution for mapping used in robotics is the occupancy grid mapping technique that represents an environment as a tessellated map[1] consisting of a number of 2D squares or 3D cubes [19, 39, 40]. The technique was originally coined by Elfes, who created a probabilistic implementation capable of creating 2D or 3D maps [39]. He also developed a method of incorporating the uncertainty in measurements from sonar and stereo vision sensors by modelling their errors.

The basic principle of an occupancy grid is to discretise the world into a number of cells and assigning each a binary classification [41] where it can be either occupied or empty. To incorporate uncertainty into the representation, the classification is stored as a probability, where 1 indicates that it is definitely occupied, 0 indicates it is definitely empty and 0.5 that no information about its occupancy is available. The technique is developed specifically to map static environments that do not change, since mapping dynamic objects entails an entirely different approach where dynamic object are identified and tracked over time.

The original implementation divided the environment into equally sized blocks and stored only a probability that each contained an obstacle. This could result in a large number of superfluous homogeneous neighbouring blocks, which was improved by adaptive grid mapping. This is implemented by transforming the map into a tree-based structure, as described by Einhorn et al. [42] and shown in Figure 2.2.

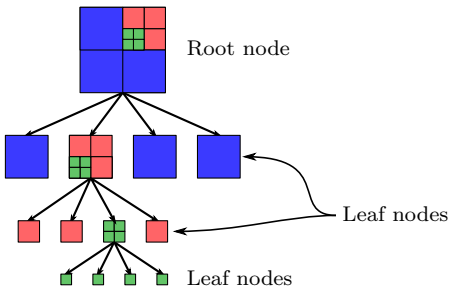


Figure 2.2: Example of a tree-based occupancy grid, showing a root node that is divided into smaller nodes.

The top level node is referred to as a root node, is the largest map cell in the tree and has the same size as the entire tree. Any node not divided into children is called a leaf node as marked in Figure 2.2, and in most implementations a probability of occupancy is calculated for these leaf nodes. Homogeneous regions can be merged by pruning sections of the tree that do not provide additional information. This improves the memory usage of the map, especially when mapping in 3D. Additionally, when some measurements

[1] This tessellation means that the map consists of a number of cells (not necessarily all the same size) that fit together without gaps or overlapping.

indicate a cell is occupied and others that it is empty, that cell may be divided into smaller cells. Each of these cells can be assigned its own probability of occupancy, which allows the map to represent complex regions in greater detail.

Occupancy grids are the obvious choice for mapping in this study, due largely to its popularity in the field of autonomous mobile robotics. Most recent projects (see Section 2.4) have used it due to its efficient memory usage and elegant implementation.

It represents the environment in a convenient way, since many navigation and collision avoidance algorithms were developed to work on such a grid-based representation. Since any point in the map can be queried directly based on the probability that it contains an object, these algorithms can easily use the map to establish the safety of specific paths.

Importantly, it offers a way to model the uncertainty in the state of a map cell, which is crucial with inaccurate sensors such as stereo vision. Mapping using occupancy grids is reliant on the inverse sensor model that is discussed in Section 2.3.

2.3 Sensor Models

The sensor model is an important part of the occupancy grid mapping technique. It describes a relationship between a map and measurements made of the environment. Sensor models are also responsible for representing measurement noise, since they affect the way that measurements relate to the environment. We update the occupancy probability of each map cell using the sensor model to find a distribution based on the new measurements, and then incorporating that distribution into the existing belief.

The forward and inverse sensor models are discussed next.

2.3.1 Forward Sensor Model

The forward sensor model is described and used by Thrun [40]. It specifies a probability distribution over sensor measurements z given a map m. It is of the form

$$p(z_t|m), \tag{2.1}$$

where z_t refers to the measurements at a certain time step and m to a map. The forward sensor model in essence describes the measurement for every possible combination of the map.

The way Thrun calculates maps using the forward sensor model involves an iterative solution. Essentially, maps that are likely to have caused a set of measurements are searched for using the expectation maximisation (EM) technique [43]. A number of other forward models exist, however, specified in different ways.

In Section 3.5 of his paper, ?? describes how he estimates the marginal posterior $P(m_{xy}, Z)$. Another problem with this algorithm is that maps are found based on the full set of measurements, making it computationally intractable to update the map every time new measurements become available. Ideally a robot should be able to calculate a map incrementally as it explores, allowing the map to be used as it is created.

An alternative way to learning occupancy grid maps with forward sensor models was proposed by Pathak et al. [44]. The authors use forward sensor models for incremental mapping, without iteratively solving maps. This approach does not explicitly calculate occupancy probabilities, and instead establishes probabilities of visibility. This has been found to be an inferior solution when compared to algorithms

that establish occupancy probabilities [45].

2.3.2 Inverse Sensor Model

The inverse sensor model, in turn, is given by

$$p(m_i|z_t), \tag{2.2}$$

and models the distribution of a map cell m_i given one time step's measurements z_t. This is not the causal direction since the map cells' states of occupancy are not caused by the measurements, which is what gives the inverse sensor model its name.

The inverse sensor model was created as part of the occupancy grid mapping technique by Elfes [46], and extended by Thrun [17]. A number of alternative models have also been suggested [47,48]. All these models have been used to create occupancy grid maps incrementally, usually using close range sensors such as sonar or lasers.

However, a principled derivation is not available for the inverse sensor model, as the available models seem to be chosen empirically. In general, the existing models are also limited by an assumption of unimodal Gaussian noise.

Due to the reasons discussed here, we focus on the inverse sensor model rather than the forward sensor model. We discuss it in detail in Chapter 5, where we also provide a derivation.

2.4 Related Projects

Using stereo vision as an input to mapping has recently received considerable academic attention, since advancements in processing power improved the image processing possible on mobile robots. A number of prominent examples from the literature are discussed here.

Some projects focus on mapping, for instance the Octomap occupancy grid mapping implementation [49] that offers a common framework for 3D mapping. Although it offers an efficient and convenient way to create maps, it is limited by an ideal sensor model that does not incorporate the uncertainty in measurements. Alternatives to the Octomap framework include the work done by Schauwecker and Zell [50], who employ a linear sensor model. Although an improvement upon the modelling in Octomap, this is inadequate for the measurement uncertainty in most practical sensors.

Most of the projects that implement mapping on board a mobile robot limit it to 2D occupancy grid maps [15,51,52]. Although sufficient for simple robot models that only consider planar movement, this can be problematic in more complex environments or with vehicles capable of moving in more than two dimensions.

The u-disparity plane [51] can be used to simplify the integration of stereo data into the global map. This is done by effectively reducing the dimensionality of the disparity map before integrating it into a 2D map. Although very efficient and useful for navigation, since large areas around obstacles are considered occupied, this technique is only viable for simple environments.

An alternative to 2D mapping is creating a 2.5D (also known as occupancy-elevation) grid map as done by Souza and Maia [53]. Instead of assigning an occupancy to each cell in a 2D grid, a Gaussian distribution is created that describes the heights of obstacles in that cell. Although this is much simpler and faster to calculate than a full 3D occupancy grid map, it contains significantly more information about obstacles than the standard 2D representation, which can be useful for navigation.

Apart from some projects that employ simple sensor models [54, 55], one of the first studies to successfully use stereo vision as input for a 3D map was done by Andert [47]. A novel sensor model is used to create 3D maps in adequate detail to autonomously navigate a remote controlled helicopter that is used in testing. However, this sensor model is provided without a principled derivation, and is limited to zero-mean Gaussian sensor noise.

Some modified versions of Andert's sensor model exist, such as the work done by Heng et al. [56], where the model is forced symmetrical around the measurement with a quadratic exponential. This is done to assign an equal occupancy probability to grids at equal distances around the measurement, without clear justification. As shown by Matthies and Shafer [57], however, the sensor model should be asymmetrical around the measurement location due to the exponential nature of stereo sensor error over distance.

2.5 Detailed Problem Statement

The problem that we focus on is creating reliable 3D maps using stereo vision as a sensor. A survey of the literature indicated that such a system is not available, with most of the existing implementations focusing on real-time calculation. There is often a direct trade-off between execution speed and accuracy. Many implementations make strict assumptions about the accuracy and range of the sensor, which puts strong limitations on the mapping system.

Using stereo vision as a sensor for mapping entails processing the images with a stereo correspondence algorithm, the results of which are known to be noisy. Triangulating these points to real world coordinates can cause more distant measurements to be very inaccurate, which can be problematic if the uncertainty in these measurements is not considered carefully.

Therefore, we wish to create a way to characterise the measurement errors made by a stereo correspondence algorithm. This characterisation should describe the error distribution, which must be incorporated into the sensor model so measurements can be reliably integrated into the map.

Since existing sensor models are generally limited to zero-mean Gaussian noise, we wish to derive an inverse sensor model that is capable of describing more complex sensor noise distributions. Considering the complexity of the stereo correspondence problem, we need a sensor model that is capable of capturing the uncertainty in such measurements. If such a sensor model is not available, it can lead to overconfidence in the calculated map.

We will follow a principled approach to deriving the inverse sensor model, ensuring that the assumptions we make are clear and physically meaningful. This should result in an inverse sensor model that is not chosen empirically, but instead comes from sound statistical principles.

Chapter 3

Stereo Vision

Retrieving distances to visible objects from a stereo camera sensor requires a solution to the correspondence problem. Due to the cameras and the rig itself being non-ideal some calibration and correction need to be done before this is possible. Even then, a correspondence algorithm may return erroneous disparities that must be characterised so they can be used reliably in the construction of a map.

Before the disparity errors are considered, a brief explanation of the geometry involved in stereo vision is provided in Section 3.1. After this, some of the issues faces by correspondence algorithms are discussed in Section 3.2. Lastly, the error characteristics of these algorithms are investigated in Section 3.3.

3.1 Stereo Vision Geometry

Most techniques used to solve the correspondence problem in stereo vision rely on a simplified model of the geometry. Before these techniques can be implemented the images need to be transformed to fit this model.

This process is detailed here by first explaining the concept of homogeneous coordinates, and then detailing the pinhole camera model. The relevant mathematics for projecting points onto the images is included, accompanied by a short description of how a camera can be calibrated. After this, we briefly describe how points can be triangulated to world coordinates, and how we rectify stereo images to simplify triangulation.

Most of the theory in this section is based on work by Bradski and Kaehler [34] and Hartley and Zisserman [58].

3.1.1 Homogeneous Coordinates

In homogeneous notation, $[x_1\ x_2\ x_3]^T$ represents not only a single vector, but a collection of scaled vectors $k[x_1\ x_2\ x_3]^T$ where $k \neq 0$. When all these classes are combined it forms the projective space $\mathbb{P}^2$, and allows for the representation of infinities in the far field. This space $\mathbb{P}^2$ is the set of lines in $\mathbb{R}^3$ passing through the origin.

Any 2-dimensional vector $[x_1\ x_2]^T$ can be written in homogeneous coordinates as $[x_1\ x_2\ 1]^T$ by projecting them onto the $z = 1$ plane, and a homogeneous vector $[x_1\ x_2\ x_3]^T$ can be written in $\mathbb{R}^2$ as $[\frac{x_1}{x_3}\ \frac{x_2}{x_3}]^T$ as long as $x_3 \neq 0$. This also extends to a 3-dimensional vector in $\mathbb{R}^3$ that can be written as a homogeneous vector in $\mathbb{P}^3$.

3.1.2 Pinhole Camera Model

The pinhole camera model is fairly simple, and is used in almost every computer vision system that requires a mathematical model for camera projection. It describes how points are projected onto an image, and is integral to the geometry of stereo vision. A brief summary is given here, and the reader is referred to the work by Hartley and Zisserman [58] for more information.

The main objective of the pinhole camera model is to project a point in 3D onto an image plane. This projection is shown in Figure 3.1, which details the projection of a point in camera coordinates $\underline{\mathrm{x}}_c = [x_c \ y_c \ z_c]^T$ through the camera centre $\underline{\mathrm{C}}$ onto an image plane with origin at the principal point $\underline{\mathrm{p}}$. It also shows the world coordinate system, which differs from the camera coordinate system by some rotation and translation. This geometry is also commonly drawn with the image plane between the camera centre and the point being projected.

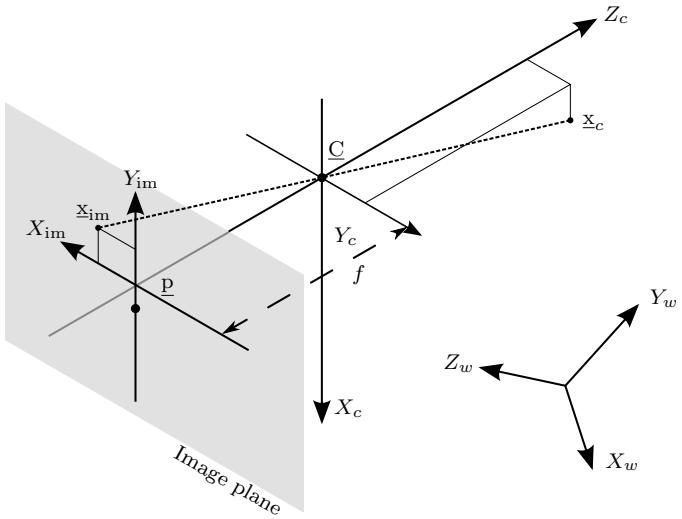


Figure 3.1: Illustration of the pinhole camera model, showing a point in camera coordinates projected onto the image plane.

The projection from the point in world coordinates ($\underline{\mathrm{x}}_w = \mathrm{R}^T\underline{\mathrm{x}}_c + \underline{\mathrm{C}}$ for some fixed rotation matrix R that relates the camera coordinate frame to the world coordinate frame) onto the image plane is described in homogeneous coordinates with the equality

$$\underline{\mathrm{x}}_{\mathrm{im}} = \mathrm{P}\underline{\mathrm{x}}_w. \tag{3.1}$$

Here P is the projection or camera matrix, of the form

$$\mathrm{P} = \mathrm{KR}[I \ | -\widetilde{\underline{\mathrm{C}}}], \tag{3.2}$$

where $\widetilde{\underline{\mathrm{C}}}$ is the Euclidean form of the camera centre in world coordinates. The matrix K is called the calibration matrix and contains the intrinsic parameters of the camera. It is of the form

$$\mathrm{K} = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.3}$$

with α_x and α_y scaled versions of the focal length that also account for scaling between world coordinates and pixels, x_0 and y_0 the pixel coordinates of the principal point (where the camera's Z-axis intersects the image plane), and s a skewness factor.

Equation 3.1 provides a way of projecting a point in world coordinates onto the image plane, given the intrinsic parameters described in K and the extrinsic parameters described by R and $\underline{\widetilde{C}}$.

3.1.3 Calibration

When we use the pinhole camera model for stereo vision, we rely on two camera matrices P and P$'$, where P is the projection matrix for the left camera and P$'$ the right camera's projection matrix. Together these matrices encapsulate the intrinsic and extrinsic parameters of the two cameras. These values are not only dependent on the cameras, but also on how they are set up relative to each other and to the vehicle.

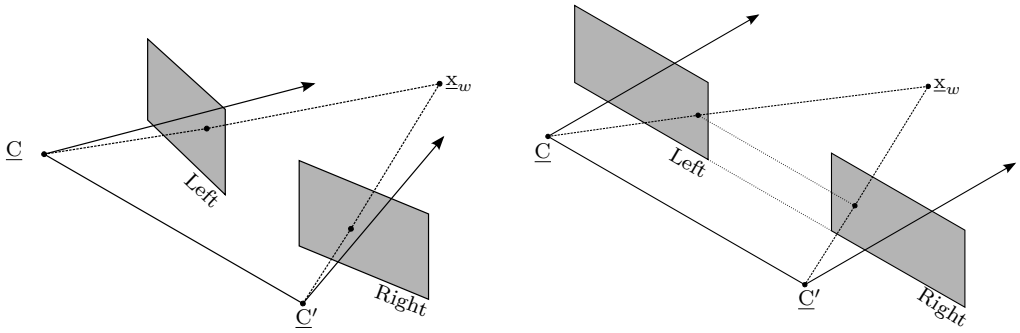
Calibration of the stereo vision sensor amounts to finding P and P$'$. This process and its automation have been detailed in the literature [34]. It usually involves capturing images of an object with known dimensions (often a planar checkerboard).

3.1.4 Triangulation

The main purpose of stereo vision is to triangulate a point to 3D world coordinates from its projection on two image planes. This can be performed by solving for $\underline{x}_w$ from the homogeneous equations $\underline{x}_{\text{im}} = P\underline{x}_w$ and $\underline{x}'_{\text{im}} = P'\underline{x}_w$, given $\underline{x}_{\text{im}}$ and $\underline{x}'_{\text{im}}$ [59]. Instead of following that route, it is better to first transform the two images through a process called rectification. This simplifies not only triangulation but also the correspondence problem , since corresponding points in the two images will be in the same row.

3.1.5 Rectification

Triangulation and indeed the correspondence problem can be simplified by a process called rectification, which ensures that matching points in the two images are in the same image row. An example of this is shown in Figure 3.2, where two stereo image planes are rectified to align the image rows.



(a) (b)

Figure 3.2: Rectification of stereo image planes, showing (a) the original image planes and (b) the rectified ones.

Provided both camera matrices

$$\begin{aligned} \mathrm{P} &= \mathrm{KR}\left[\mathrm{I} \mid -\widetilde{\underline{\mathrm{C}}}\right] \text{ and} \\ \mathrm{P}' &= \mathrm{K}'\mathrm{R}'\left[\mathrm{I} \mid -\widetilde{\underline{\mathrm{C}}}'\right] \end{aligned} \tag{3.4}$$

are available, we define new projection matrices

$$\begin{aligned} \mathrm{P}_n &= \mathrm{K}_n\mathrm{R}_n\left[\mathrm{I} \mid -\widetilde{\underline{\mathrm{C}}}\right] \text{ and} \\ \mathrm{P}'_n &= \mathrm{K}_n\mathrm{R}_n\left[\mathrm{I} \mid -\widetilde{\underline{\mathrm{C}}}'\right]. \end{aligned} \tag{3.5}$$

Here the two cameras not only have the same intrinsic camera parameters (K_n), but their orientations relative to the world coordinate system (R_n) are also the same. The only thing that differentiates the two camera matrices is that they have different camera centres in world coordinates ($\widetilde{\underline{\mathrm{C}}}$ and $\widetilde{\underline{\mathrm{C}}}'$).

We choose K_n as the average $0.5(\mathrm{K} + \mathrm{K}')$, and determine the rotation matrix R_n aligned with the vector between the two camera centres ($\widetilde{\underline{\mathrm{C}}}' - \widetilde{\underline{\mathrm{C}}}$).

The two captured images are transformed with the homographies

$$\mathrm{T}_1 = \mathrm{K}_n\mathrm{R}_n\mathrm{R}^T\mathrm{K}^{-1} \text{ and } \mathrm{T}_2 = \mathrm{K}_n\mathrm{R}_n\mathrm{R}'^T\mathrm{K}'^{-1}. \tag{3.6}$$

When these transformation matrices are applied to the respective images, it results in two rectified images with aligned image rows.

This means that any point in space that is triangulated to two rectified image planes yields a communal y_{im} coordinate, and two distinct x_{im} coordinates. The difference between these x_{im} values (the disparity) can be used to triangulate the rectified image coordinates back into world coordinates.

3.1.6 Simplified Triangulation

The required values for triangulating a point $[x_n\ y_n\ z_n]^T$ into the rectified camera coordinates are the projected locations $[x_{\mathrm{L}}\ y]^T$ and $[x_{\mathrm{R}}\ y]^T$, the baseline (distance between the two camera centres) b and the rectified focal length f of the cameras. From similar triangles in Figure 3.3 it is known that

$$\frac{f}{x_{\mathrm{L}}} = \frac{z_n}{\frac{b}{2} + x_n} \qquad \text{and} \qquad \frac{f}{x_{\mathrm{R}}} = \frac{z_n}{x_n - \frac{b}{2}}, \tag{3.7}$$

where the coordinates x_{L} and x_{R} are the projected horizontal pixel locations. Introducing the principal point, which is the pixel offset of the optical axis on the image plane yields

$$x_n = \frac{(x_{\mathrm{L}} - p_x)b}{x_{\mathrm{L}} - x_{\mathrm{R}}} - \frac{b}{2}, \qquad y_n = \frac{(y - p_y)b}{x_{\mathrm{L}} - x_{\mathrm{R}}} \qquad \text{and} \qquad z_n = \frac{fb}{x_{\mathrm{L}} - x_{\mathrm{R}}}. \tag{3.8}$$

3.2 Common Sources of Errors in Dense Stereo

The stereo correspondence problem is difficult to solve, and any algorithm will inevitably make mistakes. We define these errors by subtracting the calculated disparity value ($x_{\mathrm{L}} - x_{\mathrm{R}}$) from the ground truth, both in pixels. This leads to a measurement error measured in pixels, where a positive value indicates that the ground truth value is larger and therefore that the stereo correspondence algorithm estimated

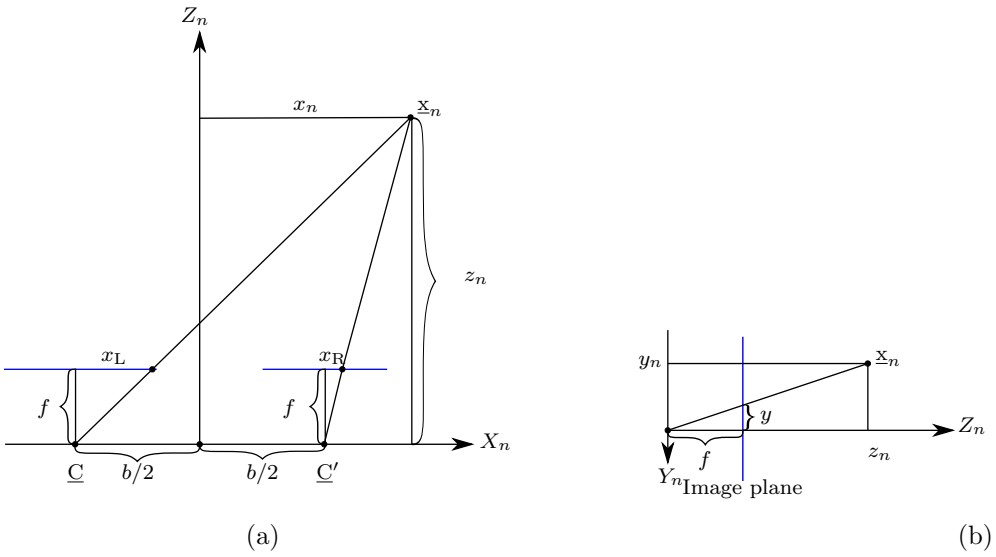


Figure 3.3: Simplified geometry used to triangulate matching points in rectified stereo images.

the object too far away.

The problem is that even disparity errors of a few pixels can be problematic, since the real world distance is inversely proportional to the pixel disparity according to Equation 3.8. Figure 3.4 shows that although a small pixel error will translate to a small error in distance at close range, the same pixel error will relate to a greater distance error if the obstacle is far away.

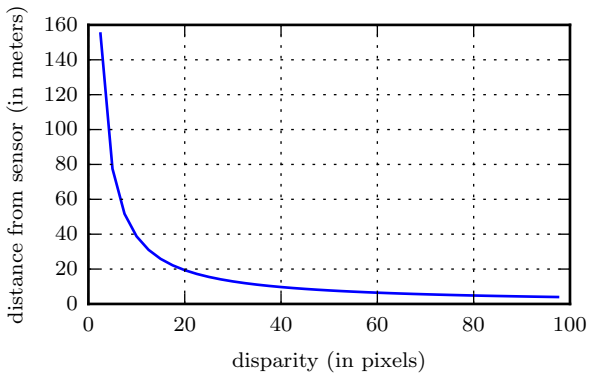


Figure 3.4: Typical inverse proportional relationship between pixel disparity and distance.

Although each stereo algorithm can have its own error characteristics, there are a number of problems that are common to all. Most of these problems occur due to the nature of the correspondence problem, while others are due to more practical aspects of solving it.

3.2.1 Bad Rectification

One possible cause of errors is inaccurate rectification. Local matching algorithms in particular rely on a strong alignment between the same rows in the two images.

This problem leads to either many points for which the correspondence problem cannot be solved in sparse techniques, or large errors if the algorithm is dense. Dense algorithms attempt to find the

correspondence of a pixel in a winner-takes-all manner, which can cause problems if a good match is not found.

3.2.2 Occlusions

Occlusions are an effect of the difference in viewing angles of the two cameras, and causes points to be visible to one camera and not the other. Stereo vision algorithms function on the premise of finding objects in the target image that are visible in the reference image, which is impossible if objects are occluded. Apart from the obvious case where one camera's view of an object is obstructed, occlusions also occur at the boundaries between objects at different distances.

It is almost impossible to identify occlusions once the stereo correspondence algorithm is complete, since the errors they cause are not predictable. One technique that has been found to be effective is dual matching [60], which involves independently matching not only the target image to the reference image, but also the reference image to the target image. It amounts to twice as much processing, but occlusions should appear as differences between the two calculated disparity maps.

3.2.3 Uniformity

Possibly the greatest challenge for outdoor stereo vision systems is lack of texturing, where uniform regions such as the sky or the walls of buildings do not contain enough information for matching. A stereo vision algorithm relies on finding corresponding pixels or regions in two images, which is impossible if the images contain large areas that are homogeneous.

The result of matching these areas can contain large errors, which can be catastrophic for mapping. Some algorithms put smoothness constraints on calculated disparities (under the assumption that nearly all scenes comprise piecewise smooth surfaces), which may limit these types of errors.

3.3 Characterisation

Different stereo correspondence algorithms work in different ways. To facilitate the accurate mapping of an environment with erroneous measurements, we need a way to model the errors in the output of a particular stereo correspondence algorithm for a given setup and environment. If this characterisation is incorporated correctly into the inverse sensor model, then the calculated occupancy probabilities should capture the uncertainty in the measurement.

To find this model, we measure the errors made by a stereo algorithm over a representative subset of data, provided that ground truth is available. We then bin these errors into a histogram to calculate the relative frequencies of different errors. However, we need a probability density function to approximate this error histogram, which we can use to set up the inverse sensor model.

A Gaussian mixture model (GMM) is a model consisting of G weighted Gaussian components that are summed, and is written as

$$p(x) = \sum_{i=1}^{G} w_i \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right), \tag{3.9}$$

where the ith component is described by its relative weight w_i, its mean μ_i, and its standard deviation σ_i. The number of components is given by G, and impacts how accurately the GMM can approximate

a given distribution. The component weights satisfy the constraints that

$$\sum_{i=1}^{G} w_i = 1 \text{ and } w_i \geq 0. \tag{3.10}$$

Given enough components, GMMs are known to be able to smoothly approximate distributions of any shape [43].

We choose to use a GMM to describe the error distribution in this study for its generality, and employ it here to approximate the error distributions of the three algorithms identified in Section 2.1.2 on two different datasets. The expectation maximisation (EM) technique [43] is used to fit GMMs with varying numbers of components to the error histogram of each algorithm on each dataset. The number of components G in the GMM is a trade-off between over-specialisation and reducing execution time with fewer components.

An important assumption we make is that the disparity error is independent of position in the image. This allows for the creation of a single model that provides a distribution for the error in every measurement from the sensor.

Each calculated GMM is shown with the histogram of the error of the respective stereo algorithm for qualitative evaluation, and details of each GMM are provided in a table. This includes the parameters for each of the calculated components, as well as the Akaike information criterion (AIC) [61] which provides a relative quality of fit to models for a given set of data. The AIC of a GMM fitted to data is defined as

$$\text{AIC} = -2\log(\text{maximum likelihood}) + 2(G), \tag{3.11}$$

and is designed to weigh a model's quality of fit against its complexity. Here the parameter G is again the number of components in the fitted GMM, and the maximum likelihood is the likelihood that the given model resulted in the data provided. Although the value itself does not have a clear physical meaning, it can be used for model selection, since a model that fits data better will have a lower AIC value.

We apply the three stereo correspondence algorithms identified in Section 2.1.2 to the synthetic Tsukuba dataset [21] and characterise the errors. Since the system is being developed for outdoor mapping, the KITTI dataset [12] is used for testing as well, since it offers rectified stereo images of urban scenes.

3.3.1 Wide Error Distributions

The EM algorithm we use to fit GMMs can be sensitive to outliers in wider error distributions. Increasing the number of components in the GMM should solve the problem with outliers, but it means that we can require many components in the GMM if the distribution contains multiple small components. In an effort to represent the original error distribution as well as possible, we fit GMMs using all available error values.

This issue is illustrated in Figure 3.5, where a GMM is fitted to a set of data. Although the first histogram suggests that the distribution consists of two components, even the GMM fitted with three components appears to be a bad fit. The part of the histogram that is outside the range shown in Figure 3.5(a) is shown in Figure 3.5(b), and indicates that although relatively small, other components exist in the distribution. Removing these values and fitting a GMM with three components to the remaining data results in the model shown in Figure 3.5(c), which appears to be a significantly better fit.

It seems that although a histogram may appear to consist of only a few components, we may require

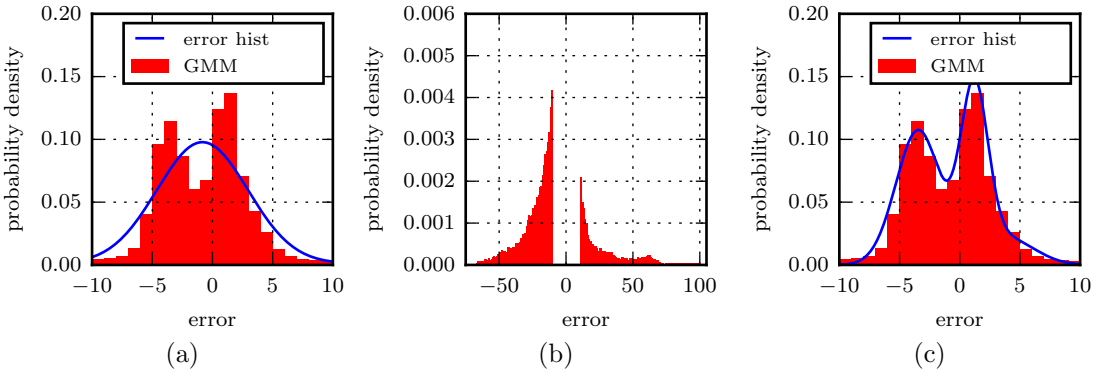


Figure 3.5: The effects of wide distributions on fitting a GMM with EM, (a) shows the original 3-component GMM, (b) the wider values and (c) a model fit only to the data shown.

many more components when fitting a GMM with EM. For example, a 7-component GMM is required to visually represent the data presented in Figure 3.5. The similarity between the AIC values of the 7-component model and the 3-component model suggests that they offer similar fits.

When we provide the parameters of GMMs in tables in this thesis, we describe the models fitted with one, two, and three components, as well as the first model that visually resembles the data (in the example case this would be the 7-component model). The last model is represented using only its components that have weights of more than 0.1.

3.3.2 Synthetic Data

The synthetic stereo dataset from Tsukuba University [21] is chosen for testing, since it contains calibrated and rectified stereo images as well as ground truth for the dense disparity maps. The first 100 pairs of stereo images from the collection of 1800 are used for characterisation, their measurement error values are collected for a specific stereo algorithm, and a number of GMMs are fitted. The number of error data points collected from these images vary from 6.9 million for block matching to 20 million for denser algorithms such as the semiglobal block matching algorithm.

The scene in this dataset is the inside of a simulated office building, starting at the familiar bust from the Middlebury set [62] shown in Figure 3.6(a). Since the dataset is computer generated it contains many areas that are not textured, which could cause problems for the correspondence algorithms.

The ground truth is available and an example frame is shown in Figure 3.6(b), with darker areas indicating objects nearer to the sensor. The ground truth contains disparity values that cannot be calculated by any stereo algorithm (areas with occlusions or no textures). This should favour sparser algorithms, provided they are capable of correctly identifying such areas as incalculable. The result of the three stereo correspondence algorithms is shown in Figure 3.7.

3.3.2.1 Block Matching

The first technique for which the characterisation is tested is block matching, which calculates fairly sparse disparity maps. The histogram of its errors is shown in Figure 3.8, with GMMs fitted with varying numbers of components. The block matching algorithm appears to calculate accurate disparity values, but the single component GMM is badly affected by the problem in Section 3.3.1. The parameters for the models are shown in Table 3.1, and indicate that each of the fitted models consists largely of a

(a) (b)

Figure 3.6: Sample image from the Tsukuba dataset [21] with (a) from the left camera and (b) the corresponding ground truth disparity map.

(a) (b) (c)

Figure 3.7: Disparity maps with differently scaled intensities for the Tsukuba dataset. (a) shows block matching, (b) semiglobal block matching, and (c) variational matching.

single component, and that other components are negligibly small.

The models fitted to the errors produced by the block matching algorithm are dominated by a single Gaussian component with standard deviation around 1 pixel. It suggests that for this algorithm the existing sensor modelling techniques mentioned in the literature study would be adequate, although a small mean offset is present that they would be incapable of representing. Due to the increased tightness of fit that the third component introduces, the 3-component GMM should be the best option.

3.3.2.2 Semiglobal Block Matching

The second technique is the more global and more complete semiglobal block matching.

Resulting characterisation is shown in Figure 3.9. The errors seem to be largely around zero, with

G	AIC	μ_1	σ_1	w_1	μ_2	σ_2	w_2	μ_3	σ_3	w_3
1	5.33×10^7	-0.80	4.04	1.00						
2	3.00×10^7	-0.38	1.07	0.92	-5.67	20.68	0.09			
3	2.75×10^7	-0.35	0.64	0.86	-1.48	4.49	0.11	-10.28	31.16	0.03

Table 3.1: Parameters of block matching GMM for the Tsukuba dataset, with G the number of components

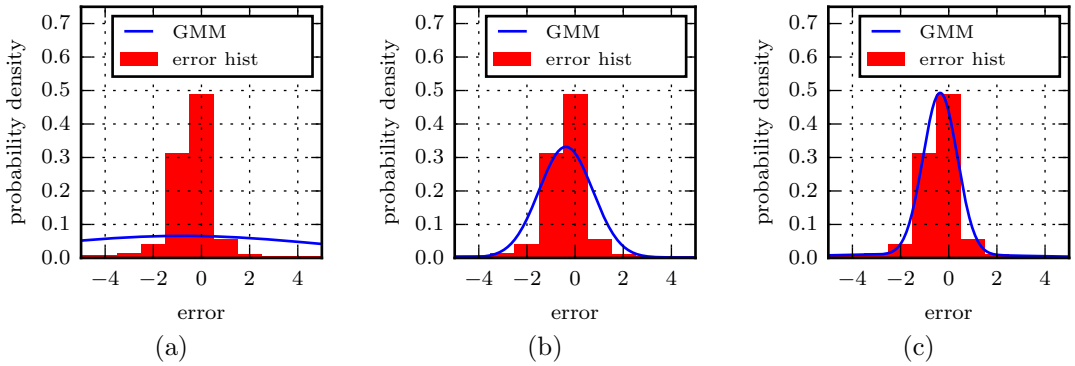


Figure 3.8: Error distribution from the block matching algorithm on the Tsukuba set. (a), (b) and (c) show GMMs with one to three components respectively. Note that only the section with non-negligible sized components is shown.

a standard deviation of roughly 1 pixel. A 2-component GMM seems to fit the data fairly accurately. This is further shown in Table 3.2, where the parameters for each of the calculated GMMs are detailed. There is a dominant component with zero mean and a standard deviation of roughly 0.7 pixels.

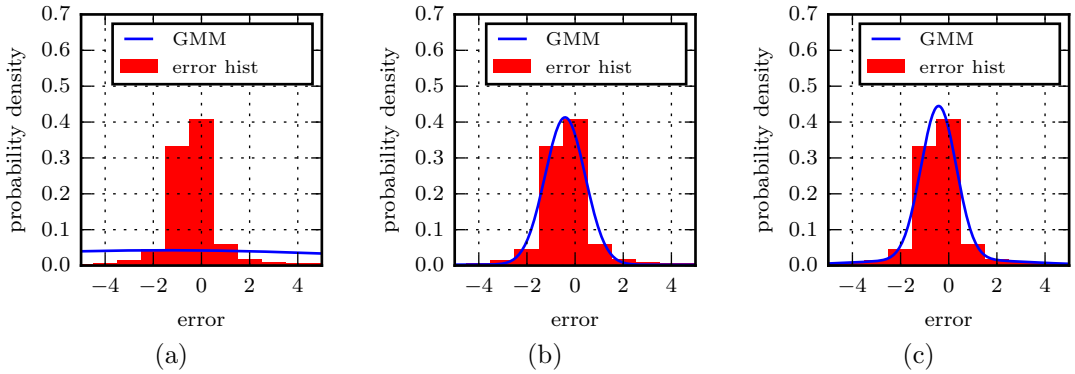


Figure 3.9: Error distribution from the semiglobal block matching algorithm for the Tsukuba set. (a), (b) and (c) show GMMs with one to three components respectively. Note that only the section with non-negligible sized components is shown.

The error distribution of the semiglobal block matching algorithm could also be represented by a single component, although it benefits from a second component due to the presence of a small but very wide component in the distribution. Although the 2-component and 3-component GMMs offer similar fits, the GMM with $G = 3$ seems to be the best option.

3.3.2.3 Variational Matching

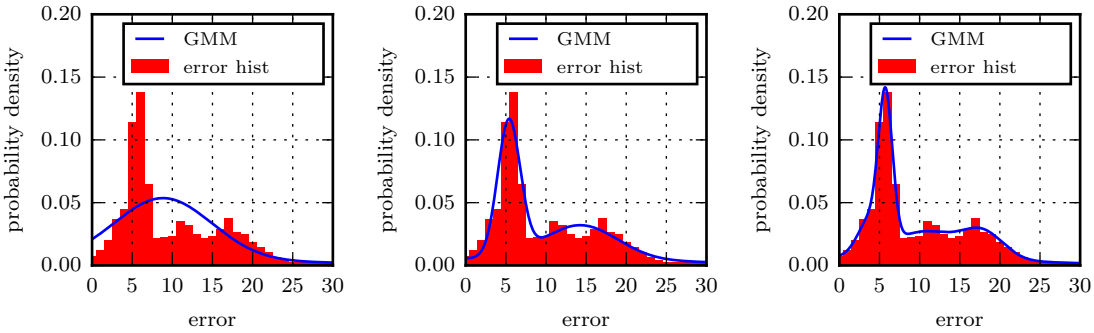
The variational algorithm offers dense disparity maps that can be calculated very quickly but with the compromise of large errors being made. The resulting disparity maps are smooth due to the area-based design of the technique.

The characterisation is illustrated in Figure 3.10, showing the different GMMs fit to the error histogram. The error distribution seems more complex than that of the other algorithms, with the introduction of more components to the GMM resulting in significantly different models. The GMM with

G	AIC	μ_1	σ_1	w_1	μ_2	σ_2	w_2	μ_3	σ_3	w_3
1	1.83×10^8	−1.48	9.34	1.00						
2	1.71×10^8	−0.42	0.78	0.86	−7.74	23.50	0.14			
3	1.71×10^8	−0.42	0.67	0.78	−0.15	3.00	0.13	−12.08	28.15	0.09

Table 3.2: Parameters of semiglobal block matching GMM for the Tsukuba dataset.

nine components is included because it is the first one to distinguish between the two secondary peaks of the distribution.



(a) (b) (c)

Figure 3.10: Error distribution from the variational matching algorithm on the Tsukuba set, characterised with different GMMs. (a) (b) and (c) show GMMs fitted with two, three and nine components respectively. Note that only the section with non-negligible sized components is shown.

The calculated parameters are shown in Table 3.3, where the first four parameters for the 9-component GMM are displayed. These are the largest components of the GMM and represent 63% of the total weight, where the other components each have weights smaller than 0.1. According to how the AIC measures quality of fit, the GMMs with three and nine components are suggested to have similar qualities of fit. However, we prefer the 9-component model since it appears to be more similar.

G	AIC	μ_1	σ_1	w_1	μ_2	σ_2	w_2	μ_3	σ_3	w_3	μ_4	σ_4	w_4
1	2.50×10^8	7.35	13.75	1.00									
2	2.35×10^8	8.86	6.04	0.76	2.80	25.22	0.24						
3	2.26×10^8	5.38	1.38	0.38	14.33	4.59	0.32	2.55	22.78	0.30			
9	2.24×10^8	5.75	0.80	0.24	17.88	2.57	0.14	11.59	3.67	0.13	9.09	12.33	0.12

Table 3.3: Parameters of variational matching GMM for the Tsukuba dataset

The Gaussian mixture model seems to be capable of representing the error distributions for all three stereo algorithms. Although a GMM with a large number of components is required to approximate the error distribution of the variational method, these characterisations suggest that the GMM should be capable of representing the error distributions of stereo correspondence algorithms applied to a synthetic dataset.

3.3.3 Real World Data

The application of this study is mapping outdoor scenarios, which is why we also consider the KITTI dataset [12]. The errors made by the stereo correspondence algorithms on its stereo images are characterised to investigate the distribution of errors made by correspondence algorithms on real world data.

Figure 3.11 shows an example image collected on the autonomous vehicle used to capture the KITTI datasets. The cameras used are PointGrey Flea2 cameras, capturing images at 1.4 megapixels. These images are calibrated and rectified as explained in Section 3.1, resulting in usable images of around 1.3 megapixels each. The two cameras are set up with a baseline of around 54 cm and have focal lengths of 721 pixels.

Figure 3.11: An example of an image from the KITTI dataset [12] of a nearly static urban scene.

Applying the three stereo correspondence algorithms to the stereo images has the result shown in the next figures, which displays the disparity maps for block matching in Figure 3.12, semiglobal block matching in Figure 3.13 and variational matching in Figure 3.14. The results appear to be less accurate than with the synthetic dataset, which is to be expected from a real world dataset due to practical issues such as inaccurate calibration parameters, reflective surfaces and overexposure.

Since it is a real world dataset there is no ground truth available, an alternative is discussed. This is followed by the results of the characterisation.

3.3.3.1 Ground Truth

A good estimate for the correct disparities must be found against which the calculated disparity values can be compared. One way to do this is to use a general consensus method, which accepts a value as correct if most of the stereo algorithms agree on it. This is risky, however, since the general consensus may be incorrect for hard to calculate areas. Another problem is that many points may not be calculable like this, causing sparse ground truth.

Alternatively, the Velodyne HDL-64E LIDAR system on board the KITTI vehicle can be used to find accurate range values. This sensor is said to have an accuracy of 2 cm across its entire range. Since the laser data is in world coordinates, it must be projected onto the reference image plane. Figure 3.15 shows that the data is not particularly dense, providing range measurement for roughly 6.7% of the pixels.

For higher completeness the data is interpolated, avoiding errors by not interpolating neighbouring values that are too far apart. A semi-complete ground truth disparity map is then available, as shown in Figure 3.16. The patchy appearance is due to plants that violate the piecewise smoothness assumption, and the step-like road surface due to how the image histogram is equalised for printing.

The first 100 frames from the dataset are used to characterise each stereo algorithm, yielding around 6.5 million data points for a sparse algorithm such as block matching and 21 million data points for a

Figure 3.12: Disparity map calculated using the block matching algorithm on a stereo image pair from KITTI dataset.

Figure 3.13: Disparity map calculated using the semiglobal block matching algorithm on a stereo image pair from KITTI dataset.

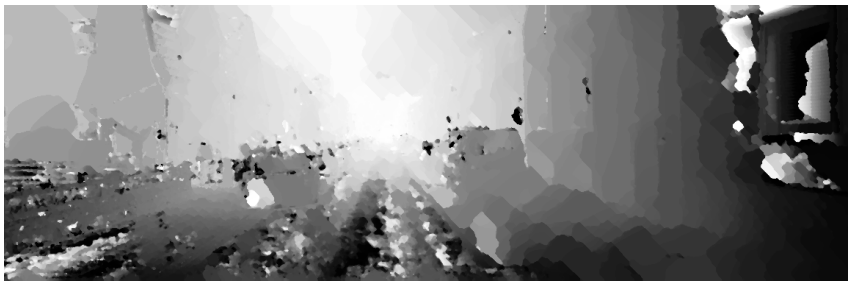


Figure 3.14: Disparity map created with the variational algorithm on a stereo image pair from KITTI dataset.

Figure 3.15: Example of LIDAR data from the KITTI dataset

Figure 3.16: Disparity map interpolated from the LIDAR data in the KITTI dataset.

more complete one like semiglobal block matching. These frames should represent typical locations to be mapped, in order to characterise the model using the correct types of errors.

3.3.3.2 Block Matching

The first algorithm whose results are shown is block matching. The differences between its disparity maps and the laser data results are collected into a histogram, as shown in Figure 3.17, and GMMs are calculated with varying numbers of components. Models with more components appear to yield GMMs that match the original data better, and the model with 7 components is the first to clearly distinguish between the two peaks in the data.

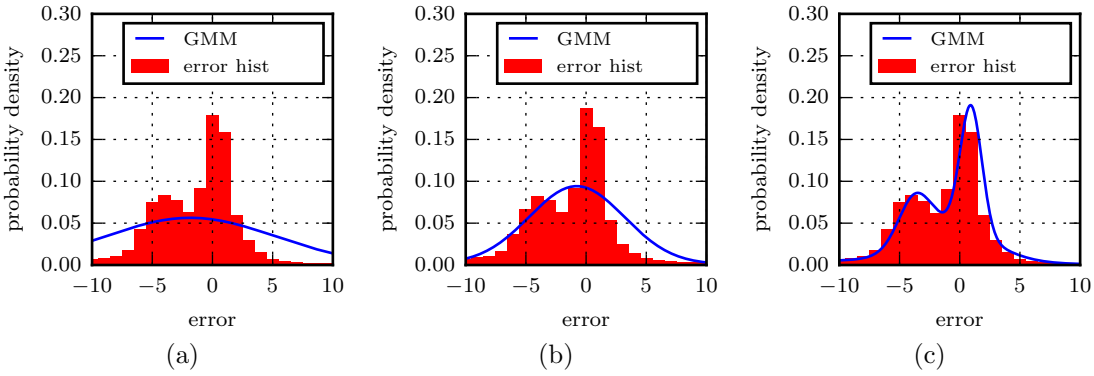


Figure 3.17: Error distribution from the block matching algorithm on the KITTI set, characterised with different GMMs. (a), (b) and (c) show GMMs fitted with one, two and seven components respectively. Note that only the section with non-negligible sized components is shown.

The calculated weights, means and variances of each of the components are shown in Table 3.4, as well as each model's AIC. Due to the models with fewer components appearing to match badly, the 7-component model that we choose is also shown as represented by its three largest components that have a combined weight of 76%.

G	AIC	μ_1	σ_1	w_1	μ_2	σ_2	w_2	μ_3	σ_3	w_3
1	4.25×10^7	−1.84	7.07	1.00						
2	3.71×10^7	−0.83	3.94	0.93	−14.45	17.62	0.07			
3	3.71×10^7	−0.84	3.95	0.65	−0.84	3.96	0.27	−14.44	17.64	0.07
7	3.53×10^7	0.93	0.85	0.35	−0.28	3.17	0.28	−3.38	1.69	0.13

Table 3.4: Parameters of block matching GMMs on the real world dataset.

Although a standard deviation for the pixel error noise of three or four pixels may seem small, it can result in large measurement error in distance when measuring further away from the sensor.

3.3.3.3 Semiglobal Block Matching

An example of a disparity map calculated using the semiglobal block matching algorithm is visible in Figure 3.13. The error distribution is shown in Figure 3.18, with fitted GMMs that have up to five

components. Again more components are required to approximate the distribution than are visible in the histogram.

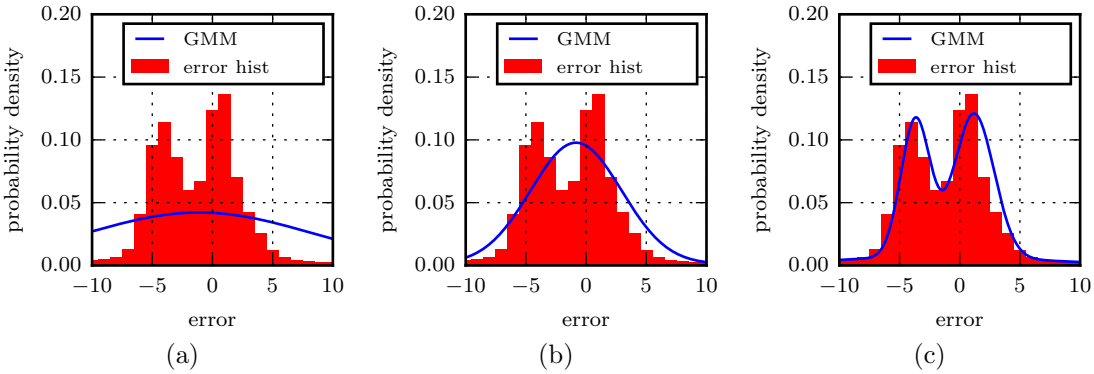


Figure 3.18: Error distribution from semiglobal block matching algorithm on the KITTI set, characterised with different GMMs. (a) and (b) show GMMs with one and two components respectively, and (c) a GMM with five components. Note that only the section with non-negligible sized components is shown.

The parameters of the matched models are detailed in Table 3.5. The 5-component GMM is represented similarly as with block matching by its three largest components, which comprise 91% of the total weight. Due to its improved fit, we prefer the 5-component GMM.

G	AIC	μ_1	σ_1	w_1	μ_2	σ_2	w_2	μ_3	σ_3	w_3
1	1.51×10^8	−1.11	9.42	1.00						
2	1.30×10^8	−1.30	2.98	0.88	−0.46	25.87	0.12			
3	1.26×10^8	−0.85	3.73	0.70	−0.86	3.75	0.22	−3.95	30.04	0.08
5	1.22×10^8	1.19	1.64	0.48	−3.68	1.18	0.33	−1.36	7.65	0.10

Table 3.5: The parameters calculated for the various GMMs to represent the semiglobal block matching's error distribution for the KITTI dataset.

3.3.3.4 Variational Matching

An example disparity map obtained with the variational method is shown in Figure 3.14. Note that the disparity map comprises of almost flat surfaces due to the area-based design of the reconstruction algorithm.

The larger errors than with the other algorithms are clear from the error histogram shown in Figure 3.19. Table 3.6 shows that the largest component is roughly 8 pixels offset, and that a second large component exists. It also shows that although the second component in the 3-component GMM is important for a good fit, the third component is negligibly small. We choose the 3-component GMM since it appears to fit the data better.

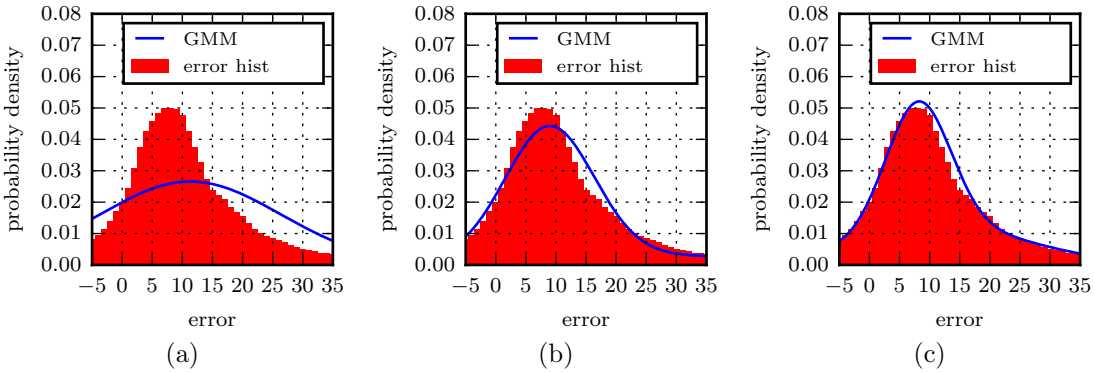


Figure 3.19: Histogram of error distribution of variational algorithm on the KITTI set, with GMMs that approximate it. (a), (b), and (c) each show a fitted GMMs with one to three components respectively. Note that only the section with non-negligible sized components is shown.

G	AIC	μ_1	σ_1	w_1	μ_2	σ_2	w_2	μ_3	σ_3	
1	2.17×10^8	11.33	15.02	1.00						
2	2.03×10^8	8.97	7.32	0.82	21.94	29.39	0.18			
3	2.02×10^8	8.11	5.33	0.55	12.55	14.01	0.39	33.77	41.12	0.06

Table 3.6: Parameters of variational GMM on the real world dataset.

Although this algorithm has larger measurement errors than the other two techniques tested, the 3-components GMM appears to fit it well. We hope that if the error distribution is well approximated, then that will make the data useful for mapping as the errors may be more predictable. This further relies on a reliable incorporation of the measurement error into the inverse sensor model, which we derive in Chapter 5.

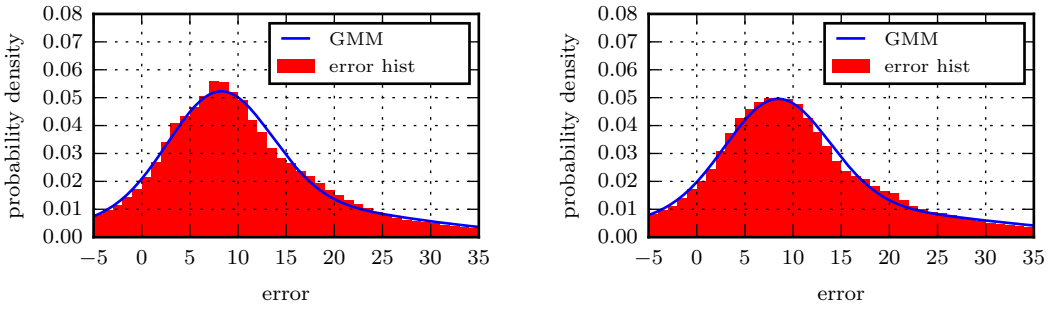
3.3.3.5 Test Data Results

It is important that the characterisation of a sensor generalises, and a different part of the dataset is characterised here to see if the models are similar. The test data is selected from a subsequent part of the dataset than the training set.

The two models fitted for the variational algorithm are shown in Figure 3.20. They appear quite similar, with the testing data appearing to not only have a similar error histogram, but also that a similar GMM is fitted to approximate it. These results suggest that the characterisations done previously should be representative of a stereo algorithm's errors for a real world dataset.

Assuming the test environment contains similar areas with similar visible characteristics, the characterisation should provide an indication of the types of errors from the stereo correspondence algorithm. If we can predict the pixel errors in the disparity values, we may have a more accurate estimate of the true ranges of the obstacles that caused the measurements. Hopefully, this will allow us to map more accurately.

The next step is to use such a characterisation of the measurement error distribution in the inverse sensor model. To do this, we need a way to incorporate a GMM into the model. This is addressed



(a) (b)

Figure 3.20: Models fitted to training and testing error distribution from the variational matching algorithm. (a) shows the model from the training set and (b) the one from the testing set.

in Chapter 5, where we describe and derive the inverse sensor model for a stereo vision sensor, in the context of occupancy grid mapping.

Chapter 4

Occupancy Grid Mapping

A popular way to represent objects in the environment of a robot is an occupancy grid map. It divides a region into separate cells and associates a probability of occupancy with each. The environment is then described by this collection of probabilities.

A definition of the map as originally developed by Elfes [46] is provided in Section 4.1, followed by a description of how measurements are incorporated into the map in Section 4.2. Lastly, an important assumption regarding independence is discussed in Section 4.3, and the implementation of the mapping technique is considered Section 4.4.

4.1 Occupancy Grid Map Definition

The occupancy grid mapping technique divides the world into cells that can be either empty or occupied. A cell is considered occupied if there is an object or part of an object within its bounds. Although the state of a cell is binary, a probability of occupancy is often associated with each cell, which allows for uncertainty in its state. The set of all the cells' probabilities of occupation can then be considered a description of the current belief of the overall state of the map.

An example of an occupancy grid map is shown in Figure 6.5, illustrating the binary occupancy of each map cell. Here again white indicates empty, black indicates occupied, and grey cells are unknown.

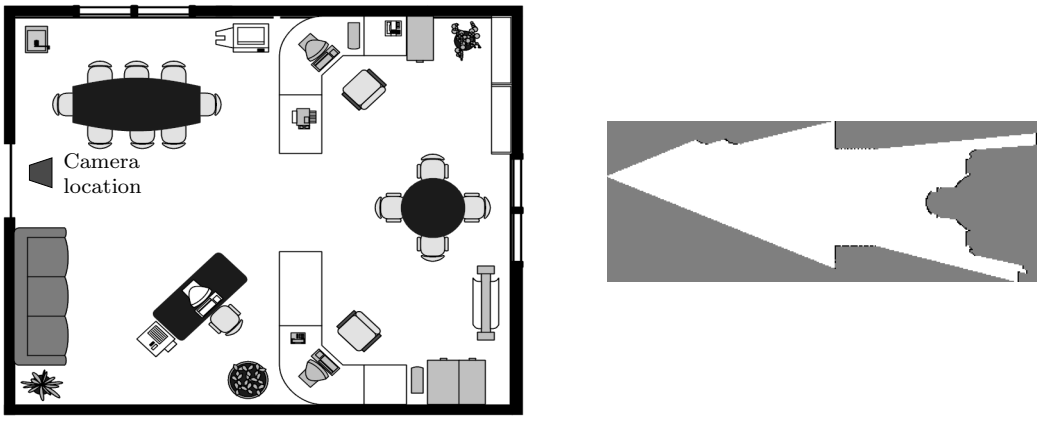
A binary variable m_i is assigned to the occupancy of cell i, and it is 1 when that cell is occupied. The probability of this cell being occupied is known as the occupancy probability and is written as $p(\mathbf{m}_i)$, with the complement given by $p(\overline{\mathbf{m}}_i)$. The goal of the occupancy grid technique is to calculate these probabilities, given information about the cells from a number of measurements.

4.2 Derivation of Update Equation

The update equation of the occupancy grid mapping technique is responsible for altering the occupancy probability of a cell using information from a measurement. This means that it incorporates the probability that a cell is occupied based on new measurements into the existing belief about the cell's occupancy. The derivation provided here for the basic occupancy grid mapping technique is based on the original work done by Elfes, extended by Einhorn et al. [42], and detailed by Thrun [17] and Joubert [19].

The general aim of the occupancy grid mapping technique is to calculate the joint distribution

$$p(m) = p(m_1, m_2, \ldots, m_N) \tag{4.1}$$



(a) (b)

Figure 4.1: An example of a binary occupancy grid map drawn of the image on the left.

for a map with N cells. This joint distribution represents the probability for each possible combination of the map, which is intractable to calculate directly due to the high dimensionality involved if N is large.

We assume that the occupancies of the cells are statistically independent and simplify this joint distribution to

$$p(m_1, m_2, \ldots, m_N) = \prod_i^N p(m_i). \tag{4.2}$$

This is a strong assumption that may not be a good approximation, but is essential to make calculation viable. See Section 4.3 for a discussion on this independence assumption.

The distribution of the map given the measurements can be written as $p(m|z_{1:t})$, where $z_{1:t}$ is the collection of measurements from various time steps $z_1, z_2, \ldots, z_t$. Each measurement includes the relevant sensor pose (often represented by $x_1, x_2, \ldots, x_t$) required for mapping.

At this stage we want to calculate the probability that a particular map cell i is occupied $p(\mathbf{m}_i)$, which forms part of the joint distribution $p(m)$. When a set of measurements is given, the probability becomes $p(\mathbf{m}_i|z_{1:t})$.

Using Bayes' rule on the occupancy probability of a cell , as well as the chain rule, yields

$$\begin{aligned} p(\mathbf{m}_i|z_{1:t}) &= p(\mathbf{m}_i|z_t, z_{1:t-1}) \\ &= \frac{p(z_t|\mathbf{m}_i, z_{1:t-1})p(\mathbf{m}_i|z_{1:t-1})}{p(z_t|z_{1:t-1})} \\ &= \frac{p(z_t|\mathbf{m}_i)}{p(z_t|z_{1:t-1})}p(\mathbf{m}_i|z_{1:t-1}), \end{aligned} \tag{4.3}$$

making the assumption that a measurement is conditionally independent of previous measurements given the map, which is related to the assumption of a static environment [40].

Applying Bayes' rule to $p(z_t|\mathbf{m}_i)$ and substituting the result into Equation 4.3 leads to

$$p(\mathbf{m}_i|z_{1:t}) = \frac{p(\mathbf{m}_i|z_t)p(z_t)p(\mathbf{m}_i|z_{1:t-1})}{p(\mathbf{m}_i)p(z_t|z_{1:t-1})}, \tag{4.4}$$

and similarly for when the cell is empty

$$p(\overline{\mathbf{m}}_i|z_{1:t}) = \frac{p(\overline{\mathbf{m}}_i|z_t)p(z_t)p(\overline{\mathbf{m}}_i|z_{1:t-1})}{p(\overline{m}_i)p(z_t|z_{1:t-1})}. \tag{4.5}$$

A cell's probability of being empty is assumed to be the complement of its occupancy probability[1], which means that

$$p(\overline{\mathbf{m}}_i) = 1 - p(\mathbf{m}_i). \tag{4.6}$$

Equations 4.4 and 4.5 can be combined to cancel out some difficult to calculate factors, forming the occupancy ratio:

$$\frac{p(\mathbf{m}_i|z_{1:t})}{p(\overline{\mathbf{m}}_i|z_{1:t})} = \frac{p(\mathbf{m}_i|z_t)}{p(\overline{\mathbf{m}}_i|z_t)} \frac{p(\mathbf{m}_i|z_{1:t-1})}{p(\overline{\mathbf{m}}_i|z_{1:t-1})} \frac{p(\overline{\mathbf{m}}_i)}{p(\mathbf{m}_i)}. \tag{4.7}$$

Equation 4.7 can be extended by substituting in the statistical complement to give

$$\frac{p(\mathbf{m}_i|z_{1:t})}{1 - p(\mathbf{m}_i|z_{1:t})} = \frac{p(\mathbf{m}_i|z_t)}{1 - p(\mathbf{m}_i|z_t)} \times \frac{p(\mathbf{m}_i|z_{1:t-1})}{1 - p(\mathbf{m}_i|z_{1:t-1})} \times \frac{1 - p(\mathbf{m}_i)}{p(\mathbf{m}_i)}, \tag{4.8}$$

which provides a way to incorporate a new measurement z_t into the existing belief on probability of a map cell ($p(\mathbf{m}_i|z_{1:t-1})$) using the belief based on the new measurement ($p(\mathbf{m}_i|z_t)$).

This is converted to log likelihoods:

$$\log\left(\frac{p(\mathbf{m}_i|z_{1:t})}{p(\overline{\mathbf{m}}_i|z_{1:t})}\right) = \log\left(\frac{p(\mathbf{m}_i|z_t)}{p(\overline{\mathbf{m}}_i|z_t)}\right) + \log\left(\frac{p(\mathbf{m}_i|z_{1:t-1})}{p(\overline{\mathbf{m}}_i|z_{1:t-1})}\right) - \log\left(\frac{p(\mathbf{m}_i)}{p(\overline{\mathbf{m}}_i)}\right) \tag{4.9}$$

to allow for increased accuracy near the probability boundaries (near 0.0 and 1.0), and to simplify map updating from a product to a sum. This log likelihood can be converted to a probability, and therefore the normal probability format and its log likelihood will be considered interchangeable in this study.

In Equation 4.9 the left-hand term describes the probability of the cell being occupied, the term on the right hand side is the new information, followed by the previous information in the third term and the prior belief in the last term. This provides a way to incrementally update a cell's occupancy probability with new information, given the occupancy probability that is based on the new measurement ($p(\mathbf{m}_i|z_t)$). This probability is referred to as the inverse sensor model. To update the map with a new measurement, we calculate this probability for each map cell using the inverse sensor model, and then use the update equation to incorporate it into the existing belief of each cell's occupancy. This is the essence of incremental mapping, which allows the integration of new information without recalculating the entire map.

4.3 Independence Assumption

The assumption of independence between map cells is at the core of the occupancy grid mapping technique, and allows for a simplification essential to incremental updating. This assumption says essentially that the state of occupancy of each map cell can be calculated without considering the states of the other cells.

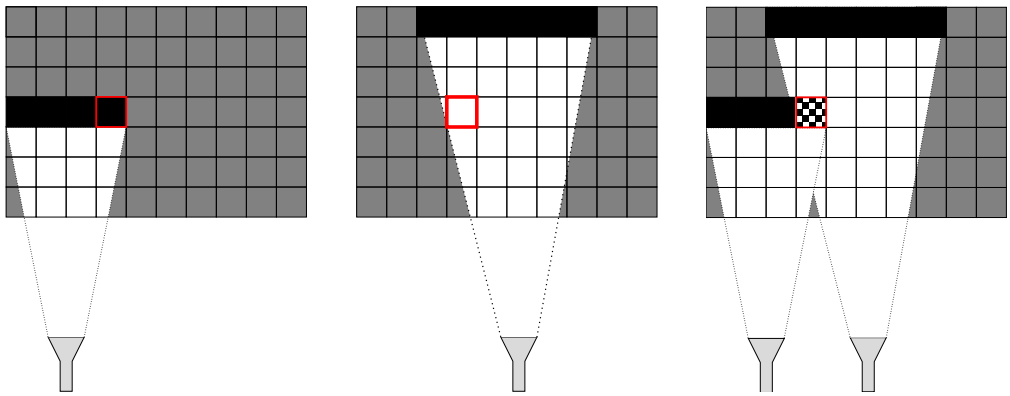
[1] This depends on the definition of the occupancy of a cell, which here refers to whether an object or part of an object exists within the bounds of a map cell [17].

The overall probability of the map occupancy can therefore be reduced as in Equation 4.2 to a conditional distribution where

$$p(m_1, m_2, ..., m_N|z_{1:t}) = \prod_i^N p(m_i|z_{1:t}). \tag{4.10}$$

There are two scenarios where this is not a valid assumption: where the measured beam is wider than a single cell, and where the range uncertainty is not limited to a single map cell.

The first case is a common problem with sensors that take measurements covering a wide beam, since this creates dependencies between the different cells in the beam. A clear example of this is when a map cell is in the measurement beams of two wide measurements and the map is updated assuming independence between cells, as shown in Figure 4.2.



(a) (b) (c)

Figure 4.2: An illustration of the violation of the independence assumption under large angle uncertainty. Two conflicting measurements are shown in (a) and (b), causing an apparent problem in (c) in the marked cell. Redrawn from Thrun [40].

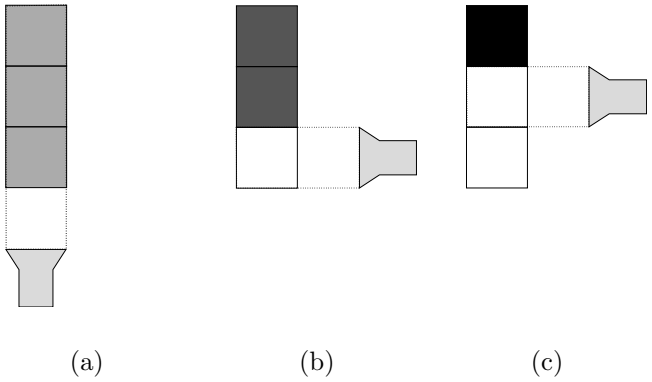
In this example, the probability of occupancy for the highlighted cell is increased by the first measurement, since it suggests that an object exists along an arc at the measurement distance. However, the second measurement beam suggests that this map cell is empty, which reduces its probability of occupancy. This creates conflicting information about cells overlapped by two measurement beams.

A measurement in a wide beam means simply that an obstacle exists somewhere along its arc, but the occupancy probabilities of all the cells in the beam are updated independently. In truth the two measurements are not contradictory, as the obstacle that is measured in Figure 4.2(a) may be outside of the measurement beam in Figure 4.2(b) and therefore the conflicted cell should actually be marked empty. This independence between cells causes the ambiguity of conflicting measurements [17], but keeping all of the dependencies between the map cells would require an impractical amount of maintenance. Essentially if all the dependencies were to be considered, the occupancy probability of each map cell would be a function of the occupancy probability of every other cell. With a large map this leads to very large dimensionality in the calculation.

With stereo vision, however, the beam widths are very thin due to the density of the pixels on the image planes (see Section 4.4.1). This makes the independence assumption acceptable under the condition that the cells are considerably larger than the beam widths, making it unlikely that a single

beam would provide information on multiple cells.

The second case is where a substantial uncertainty exists in range, but a small uncertainty in angle, which creates a dependency as shown in Figure 4.3. In Figure 4.3(a) a sensor measurement indicates that an object is present in one of three segments, with Figures 4.3(b) and 4.3(c) establishing that the bottom two are empty. In this example the dependencies between the different map cells are considered when the measurements are integrated into the map. The occupancy probability of each cell is represented by its colour, with black indicating definitely occupied and white definitely empty. Uncertain cells are indicated with shades of grey, with a darker colour indicating more likely to be occupied.



(a) (b) (c)

Figure 4.3: A sensor with range uncertainty measures an obstacle in one of three map segments with equal probability in (a). Further measurements in (b) and (c) show that the first two segments are empty, and therefore intuitively that the top segment must be occupied.

Intuitively it is clear that the obstacle is in the top cell of Figure 4.3, even though no information has been gleaned about it directly since the first measurement. The uncertainty in the first measurement about which map cell the object is in creates a dependency, since the occupancy probability of the top cell is influenced by measurements that do not observe it.

Due to their relatively large range uncertainty, stereo vision measurements create significant dependencies. With the implementation of the occupancy grid mapping technique this reduces the quality of occupancy probabilities, since the dependencies are ignored. We must make this independence assumption, however, in order to make calculating the probabilities of occupancy tractable, since without it we cannot use the update equation as shown.

4.4 Implementation

A number of issues with implementing the occupancy grid mapping technique are described here, mostly involving the incorporation of new measurements. The assumptions made regarding the measurement beam, the concept of ray casting, and the Octomap library [49] are discussed here.

4.4.1 Measurement Beam

Although a measurement typically consists simply of a range value and angle, it represents a beam from the sensor to the measurement origin. In the simplest case, a measurement implies that the space within the beam between the sensor and the measured range is empty.

When uncertainty exists not only in the measured range but also in the angle of the measurement (and the orientation of the sensor) the beam is often modelled as a cone. Sensors with large angular uncertainty such as sonar or radar lead to wide measurement cones, since the angular location of the measurement origin is uncertain.

With stereo vision sensors, however, the angle of a single pixel is minimal (as small as 0.01° for a CCD camera according to Joubert [19]) which creates a thin cone. The problem with implementing a cone measurement model is that it greatly increases the complexity of the update method. Therefore, since the tangential uncertainty of a measurement is less than 10 millimetres at a range of 5 metres, the simplification is made that the measurement cone is approximated with a ray.

This reduces the complexity of the measurement beam to a 1-dimensional measurement ray. It means that the inverse sensor model is a function of only the measurement distance, which simplifies its usage. It also simplifies finding the map segments that the beam passes through, which greatly optimises the execution time of updating the map.

4.4.2 Ray Casting

An important part of updating the map with a measurement is traversing along the measurement ray from the sensor's position to the measurement location. The basic principle is that the occupancy probabilities of all map cells that the measurement beam passes through need be updated using the update equation (Equation 4.7).

For this implementation we define each measurement as one correponding pixel pair that relates to a single 3D datapoint. This means that each pixel from the disparity map is integrated into the map individually and independently.

Figure 4.4 shows an illustration of a measurement in the process of being integrated into the map, highlighting the currently evaluated map cell (cell i). The ray is commonly sampled at the centre of each cell to retrieve the distance from the sensor to that cell, which is used to evaluate the inverse sensor model.

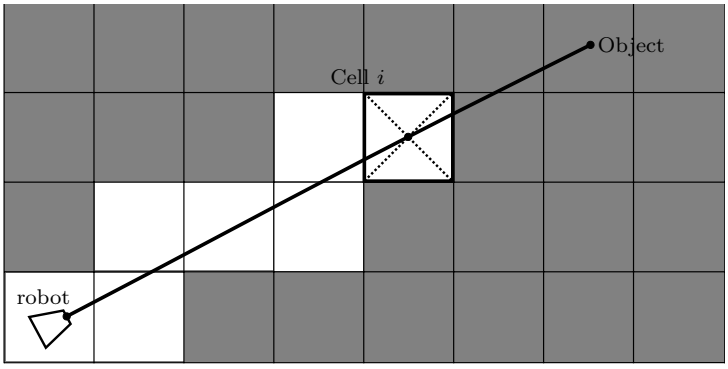


Figure 4.4: Example of ray casting, showing a 1-dimensional measurement beam being sampled at a map cell to update its occupancy probability. The cells from the robot to cell i have already been sampled and their occupancy probabilities were reduced.

With an ideal model this updating simply means that the cell where the measurement originated is marked as occupied and all of the ones between it and the sensor are marked empty. This can be equated to the ideal sensor model as shown in Figure 4.5, where the probability $p(\mathbf{m}_i|z_t)$ is shown for different

distances from the sensor. An occupancy probability of 1.0 (definitely occupied) is only assigned to cells within half a cell size (0.5 metre) from the measured distance at 5 metres.

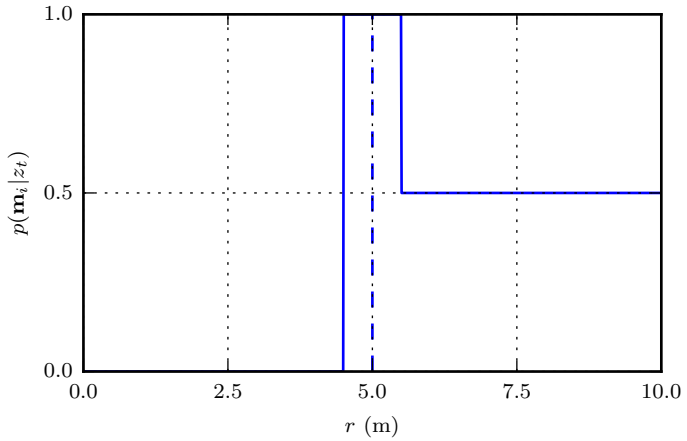


Figure 4.5: Ideal sensor model representing a noise-free measurement taken at 5 metres which dictates that only the cell (with a width of 1 metre) should be considered occupied and all cells in between empty. Cells further from the sensor than the measurement remain unknown at 0.5 probability.

4.4.3 Robot State Uncertainty

Mapping algorithms rely on an accurate estimate of the orientation and location of the robot (particularly of the sensor), otherwise it cannot perform ray casting. In real world applications uncertainty exists in the state of the robot, which needs to be handled correctly to ensure that the map is reliable.

According to Thrun [17], the uncertainty in a robot's state is described by a probability density function (PDF), which can be approximated using different techniques. One example of this is using a Gaussian PDF as in the extended Kalman filter (EKF) that is often used in simultaneous localisation and mapping (SLAM). Another option is to represent the uncertainty using a particle filter, for example with FastSLAM 1.0 or 2.0 [14]. These particles can be used to approximate any complex PDF, while EKF SLAM is limited to a unimodal Gaussian distribution.

If the uncertainty is in the form of a Gaussian PDF it is often incorporated into the measurement beam [19] by combining it with the uncertainty in the measurement. This is effective if a PDF is available for both the measurement uncertainty and the robot state, which is not always the case.

An alternative to this is to use a number of particles to represent the uncertainty in the robot state. This, however, means that for one measurement the map must be updated separately for each used particle [14, 63], which leads to longer execution times for updating the map with new measurements.

These techniques are different ways to incorporate an inaccurate robot state into a mapping algorithm, but are both very computationally intensive. Especially if the state distribution is complex, they can greatly increase the execution time of the algorithm. In this thesis most of the tests are done with data for which ground truth positional information is available and therefore no uncertainty exists about the sensor pose. The tests without ground truth must offer an accurate pose estimate, as is discussed with the results in Chapter 6. Therefore, implementing these techniques to incorporate the robot state uncertainty is considered to be outside the scope of this thesis.

4.4.4 Octomap

The implementation of the occupancy grid mapping technique is largely universal across different applications, which makes a common framework an efficient way to implement it. Such a framework offers a fast and simple way to implement a mapping system in 3D. The Octomap [49] project is a popular example of such a common framework, offering a sophisticated mapping system with many features.

Octomap describes the environment using a tree-based structure for the map, and handles the compression of unneeded nodes. It uses an ideal model for adding measurements to the map, which marks only the map cell where the measurement originated as occupied. Since the project is open-source, the source code is available and allows for modifications to the implementation. We modify the sensor model of Octomap by implementing some popular inverse sensor models from the literature and the model we derive in Chapter 5.

The Octomap library is bundled with a 3D map visualiser named Octovis that can visualise the maps calculated by Octomap. It can show empty as well as occupied areas, making it very useful for illustrating maps. All of the map visualisations shown in this thesis are generated with this tool.

Note that this visualiser draws the occupancy probabilities at discrete intervals: definitely empty, probably empty, entirely unknown (usually not drawn), probably occupied, or definitely occupied. This greatly simplifies the drawing of the map, but loses some of the information in the map. The Octomap implementation also keeps calculated occupancy probabilities within certain bounds, which means that drawn cells are not entirely white or black.

We can now represent an environment with a map, and update it using occupancy probabilities based on new measurements. We describe and derive the inverse sensor model in Chapter 5 that is used to calculate these occupancy probabilities. The model needs to capture the uncertainty in measurements that we characterised in Chapter 3, otherwise it may cause overconfidence or errors in the map.

Chapter 5

Sensor Model

The sensor model relates distance measurements from the sensor to the probability that an obstacle is present. It is used to update the occupancy grid map as explained in Section 4.2, where the probability it calculates is combined with the existing belief about the occupancy of a map cell.

Due to the noisiness of measurements, the reliability of an occupancy grid map depends on the inverse sensor model. This model is responsible for representing the uncertainty in measurements, and for ensuring that the probabilities describing the occupancy grid map are correctly updated when new measurements are incorporated.

Various approaches to the sensor model are discussed here, starting with some existing inverse sensor models in Section 5.1, followed by our principled inverse sensor model in Section 5.2.

5.1 Existing Inverse Sensor Models

The first derivation for the inverse sensor model (ISM) was done by Elfes [46] as part of the occupancy grid mapping technique update function (Equation 4.7). The probability that it calculates is

$$p(\mathbf{m}_i|z_{1:t}), \tag{5.1}$$

where $z_{1:t}$ is a set of measurements that include the relevant poses and $\mathbf{m}_i$ is the event that the occupancy of cell i has a value of 1. This probability refers to the likelihood that cell i contains an obstacle based on the information provided by the given set of measurements.

A number of different approaches have been followed to derive an appropriate ISM. Two of the most important ones will be shown here: the original as presented by Thrun [17] and Joubert [19], and the one created by Andert [47]. Although other methods exist, the presented ones are commonly used and should provide a clear explanation of the concept.

Each approach is detailed by a derivation, followed by a study of all the parameters that describe the model. In describing the parameters we also evaluate performance by applying the model to different scenarios.

5.1.1 Inverse Sensor Model by Thrun

The first derivation of the ISM is taken from work done by Elfes [46] and detailed by Thrun [40]. The process of training this type of inverse sensor model for a stereo vision sensor is described by Joubert

[19]. The derivation provided here is a combination of these three studies.

5.1.1.1 Derivation

Initially, the measurements are considered to be made without noise, which leads to the model shown in Figure 5.1. In this model, any cell closer to the sensor than the one in which the measurement originated is considered empty, around the measurement considered occupied, and unknown behind that. The model uses the size of a cell L to ensure that the cell where the measurement originated is considered occupied. A representation of this model is also shown in Table 5.1, where the distance from the sensor is given by r and the measurement is at $z_t = 5$ m.

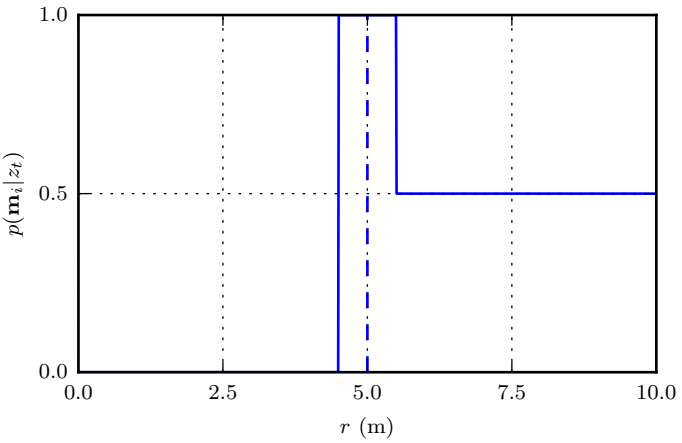


Figure 5.1: Ideal sensor model as used by Thrun in the derivation of the inverse sensor model.

Occupancy	Range	Description
0	$0 \leq r \leq z_t - L/2$	Empty in front of obstacle
1	$z_t - L/2 \leq r \leq z_t + L/2$	Occupied at obstacle
0.5	$r \geq z_t + L/2$	Unknown behind obstacle

Table 5.1: Ideal ISM occupancy probabilities.

In order to incorporate the error in measurements, a Gaussian noise model is assumed for the measurement. This means that the correct range to the nearest obstacle is assumed to be distributed around the measured distance with a standard deviation of σ, which can be written as

$$r \sim \mathcal{N}(z_t, \sigma^2). \tag{5.2}$$

The standard model for Gaussian noise used here is

$$p(r) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(r - z_t)^2}{2\sigma^2}\right). \tag{5.3}$$

It has been shown [19] that for stereo vision sensors with rectified images (see Section 3.1.5) the standard deviation σ can be approximated using

$$\sigma(z_t) = \frac{z_t^2}{bf}\sigma_d, \tag{5.4}$$

where b and f are the camera baseline and focal length respectively, and σ_d is the standard deviation of the sensor pixel disparity error. Here the approximation is made that any measurement is from the centre of the camera, where the provided conversion from measured disparity to distance is exact.

To propagate this uncertainty over the ideal sensor model the Gaussian noise distribution is convolved with it [19]. The motivation for this process is unclear, however, and it seems to be chosen for the calculation simplicity of its closed form result.

The result of this convolution is given by

$$(g * h)(r) = p(\mathbf{m}_i|z_t) = -\frac{1}{4}\text{erf}\left(\frac{r - z_t - \frac{L}{2}}{\sqrt{2\sigma(z_t)^2}}\right) + \frac{1}{4} + \frac{1}{2}\text{erf}\left(\frac{r - z_t + \frac{L}{2}}{\sqrt{2\sigma(z_t)^2}}\right), \tag{5.5}$$

which employs the error function

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^{\infty} \exp\left(-\tau^2\right) d\tau \tag{5.6}$$

and the standard deviation provided by Equation 5.4.

This function is used as the inverse sensor model to provide a probability of occupancy for each map cell along a measurement ray, given the distance from the sensor to the centre of the cell (r), the measurement range (z_t) and the sensor parameters (f, b, and σ_d). When a measurement is taken, this function is evaluated during ray casting at every map cell as discussed in Section 4.4.2 to update the occupancy probability.

5.1.1.2 Parameters

In order to study the impact that each of the parameters of the model has on the inverse sensor model from Equation 5.5, each is altered in turn. This creates a number of representative sensor models that should provide a good understanding of how the system responds to different scenarios.

Unless stated otherwise, a map cell size of 0.25 metres and a standard deviation of the disparity error distribution of 1 pixel is used. Each measurement distance used to create a model is indicated with a dashed line in the same colour as the relevant sensor model, or in black if a single distance is used for multiple models.

Parameter 1: Measurement Distance (z_t)

Since the error of a stereo vision sensor is distance dependent [19] the sensor model changes greatly for measurements at different distances. We expect that closer measurements will have higher occupancy probabilities near the measurement, since the true range of the nearest object is known with higher accuracy. Nearer measurements, however, have a higher risk of occlusions.

When implemented, Equation 5.5 yields responses as shown in Figure 5.2. Ranges close to each measurement are assigned high probabilities of occupancy, since the sensor model suggests that this is where the object that caused the measurement is located. Cells nearer the sensor are assigned lower probabilities of occupancy, since they are known to not contain obstacles. The map is left unaffected further away from the sensor than the measurement by the ISM returning an occupancy probability of 0.5, since this region is not visible.

An issue with this model is that for very distant measurements it does not yield the maximum probability of occupancy at the measured distance. This is demonstrated in Figure 5.3, where the

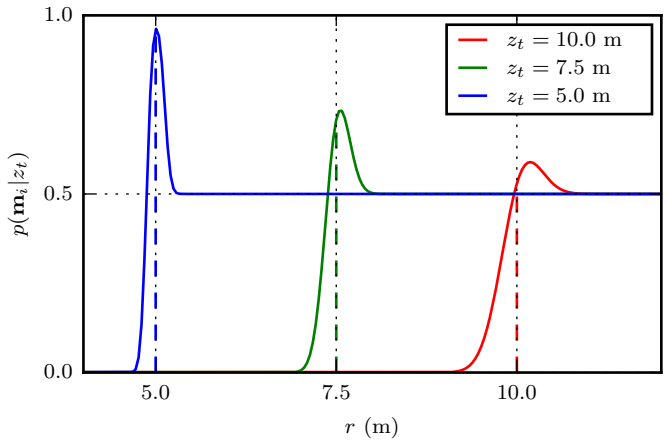


Figure 5.2: Inverse sensor model as derived by Thrun for near measurements. The distance dependent noise visibly decreases the maximum occupancy probability for distant measurements.

probability of occupancy at the measurement is approximately 0.25, which means that the model is fairly certain that there is no object in the environment at the measured distance.

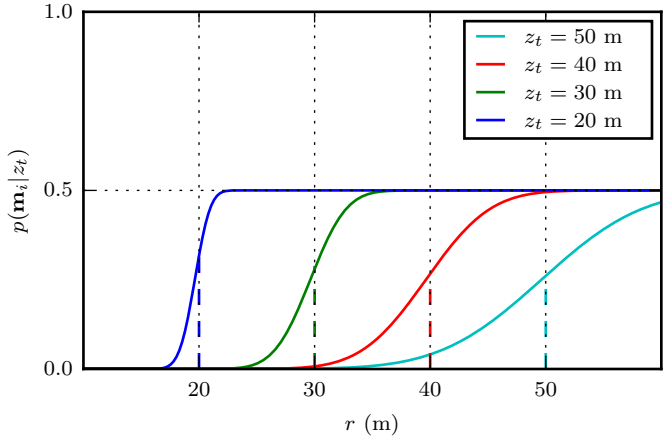


Figure 5.3: Inverse sensor model as derived by Thrun for distant measurements

This problem is clear when we consider Equation 5.5 and let z_t tend towards infinity. The standard deviation from Equation 5.4 is also substituted for values of z_t larger than r to yield

$$\begin{aligned}\lim_{z_t\to\infty} p(\mathbf{m}_i|z_t) &= \lim_{z_t\to\infty}\left(-\frac{1}{4}\text{erf}\left(\frac{r-z_t-\frac{L}{2}}{\sqrt{2\sigma(z_t)^2}}\right)+\frac{1}{4}+\frac{1}{2}\text{erf}\left(\frac{r-z_t+\frac{L}{2}}{\sqrt{2\sigma(z_t)^2}}\right)\right)\\ &= \lim_{z_t\to\infty}\left(-\frac{1}{4}\text{erf}\left(\frac{bf(r-z_t-\frac{L}{2})}{\sqrt{2}z_t^2}\right)+\frac{1}{4}+\frac{1}{2}\text{erf}\left(\frac{bf(r-z_t+\frac{L}{2})}{\sqrt{2}z_t^2}\right)\right)\\ &= \frac{1}{4},\end{aligned} \tag{5.7}$$

since $\text{erf}(0) = 0$. This shows that for very distant measurements this inverse sensor model will converge to a constant value of 0.25, and so empty the entire map along the measurement ray.

Parameter 2: Pixel Noise Standard Deviation (σ_d)

In this ISM the pixel noise standard deviation is scaled to a distance noise standard deviation with Equation 5.4, which means that greater pixel noise decreases the occupancy probabilities near the measured range in a similar way to parameter 1. This is shown in Figure 5.4, where different pixel noise standard deviations are used with a single measurement distance. It also indicates that a higher pixel noise increases the occupancy probabilities of some of the cells close to the sensor. The distance to the nearest object is less accurately known, and therefore it may be in one of these cells nearer to the sensor.

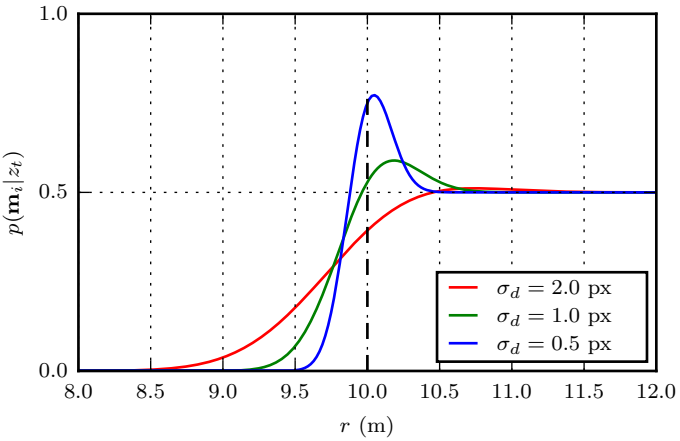


Figure 5.4: Thrun's inverse sensor model drawn with various pixel noise standard deviations. As the noise increases, occupancy probabilities near the measurement decrease.

Parameter 3: Map Cell Size (L)

The size of a map cell is incorporated into this ISM to ensure that the occupancy probability of the map cell where the measurement originated is increased. Figure 5.5 shows how different values for L influence the sensor model, indicating that a model with a larger L assigns probabilities of occupancy greater than the minimum to a wider range around the measured distance.

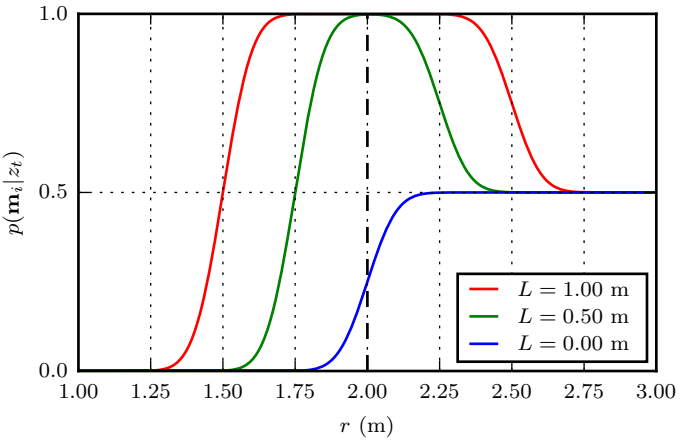


Figure 5.5: Some sensor models with varying map cell sizes (L) created using the Thrun's ISM. Measurements are at $z_t = 2$ m, and wider areas are considered occupied with greater cell size.

5.1.1.3 Conclusion

This inverse sensor model seems to work well for near measurements distances, but has problematic results for ones that are further away. Most implementations eliminate this problem by limiting the distances of measurements that are used, but some information may be lost due to useful measurements being ignored. If this distance is chosen too short, then some map cells may be updated incorrectly.

5.1.2 Inverse Sensor Model by Andert

An alternative inverse sensor model as created by Andert [47] is also presented here. It is newer than the model by Thrun, and has been used successfully to create occupancy grid maps [53].

Essentially, Andert's inverse sensor model is based on the same principles as Thrun's, where the hard decisions made in the ideal model are softened to incorporate measurement error. Andert does this by adding an exponential component to the model, which is scaled to achieve the desired transition from empty to unknown.

The different components of the function are detailed here as an explanation of the inverse sensor model it represents, since a more concrete derivation has not been published. This is followed by a discussion of the parameters that describe the model.

5.1.2.1 Derivation

The first part of this ISM to consider is an ideal model that empties everything in front of the measurement location and leaves everything behind it unknown. This is implemented by defining a function

$$P_{\text{occ}}(r) = \begin{cases} p_{\min} & \text{where } r \leq z_t \\ 0.5 & \text{where } r > z_t \end{cases} \tag{5.8}$$

where $p_{\min}$ is the minimum occupancy probability that can be assigned to an empty map cell.

The other important part of this inverse sensor model is a quadratic exponential that is scaled with a constant significance factor. It increases the probability around the measurement location with a curve that has a width relative to the uncertainty in the sensor.

Making an approximation similar to the one in Equation 5.4, the standard deviation of the pixel noise is scaled to a standard deviation of distance noise using

$$\sigma = \left(\frac{z_t}{z_c}\right) \frac{z_c^2}{bf}\, \sigma_d, \tag{5.9}$$

which employs the measurement z-coordinate in camera coordinates (z_c), the measurement distance (z_t), and the camera parameters (b and f). The scaling factor $\frac{z_t}{z_c}$ is used to combat the effects of the approximation made in scaling the standard deviations, and increases the measurement distance standard deviation when the measurement is not made at the camera centre on the image plane.

The exponential used in the model is symmetrical around the measurement location, and is given by

$$\exp\left(-\frac{1}{2}\left(\frac{r-z_t}{\sigma}\right)^2\right), \tag{5.10}$$

with σ calculated using Equation 5.9. It is scaled with

$$\frac{k}{\sigma\sqrt{2\pi}} + 0.5 - P_{\text{occ}}(r), \tag{5.11}$$

with k a constant significance factor that influences the impact of a single measurement on the map. The scaling from the combination of the significance factor and the ideal model from Equation 5.8 makes the quadratic exponential asymmetrical around the measurement, since map cells beyond the measurement are further away from the sensor and therefore have a greater measurement distance uncertainty.

The inverse sensor model as defined by Andert can now be written, in terms of the distance that the centre of a map cell is from the sensor (r), as

$$p(\mathbf{m}_i|z_t) = P_{\text{occ}}(r) + \left(\frac{k}{\sigma(z_t)\sqrt{2\pi}} + 0.5 - P_{\text{occ}}(r)\right)\exp\left(-\frac{1}{2}\left(\frac{r-z_t}{\sigma(z_t)}\right)^2\right), \tag{5.12}$$

which can be directly calculated at different ranges for a given measurement.

5.1.2.2 Parameters

To investigate the effects of the different parameters used in Equation 5.12, each of them are varied in turn. This leads to a number of models for each parameter that illustrates its influence on the system.

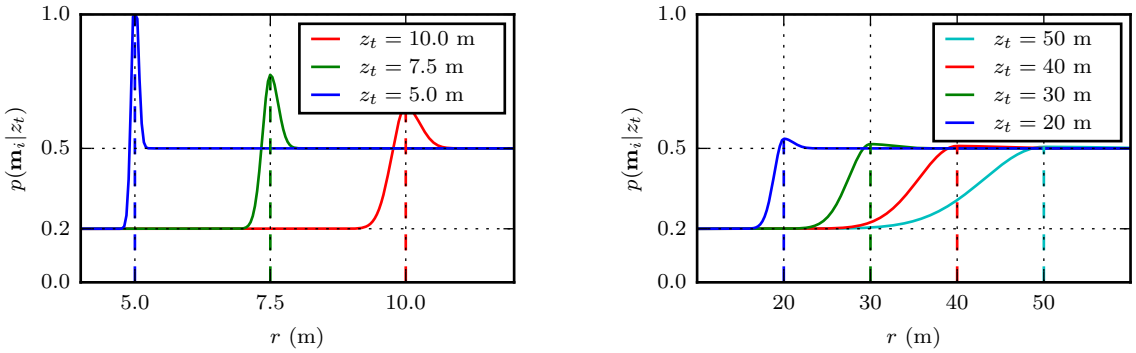
Unless stated otherwise, all models are created with a significance factor of $k = 0.1$, error distribution standard deviation of $\sigma_d = 1.0$ px and with $p_{\min} = 0.2$. As before, the relevant measurement distances for each model are marked with a dashed line.

Parameter 1: Measurement Distance (z_t)

To study how the sensor model responds to different measurement distances, some models are plotted in Figure 5.6. The closest measurement at 5.0 metres leads to high occupancy probabilities near it, while greater distances cause lower probabilities. Greater measurement distances also lead to models with wider peaks, indicating less certainty about where the measurement originated.

This model does not display the same issues with further distances in Figure 5.6(b) that Thrun's model showed in Figure 5.3. The model calculates a minimum occupancy probability of 0.5 at the measured distance, and a constant 0.5 further away. This is acceptable, since a value of 0.5 will have no effect on the map.

At the closest measurement of 20 metres the model considers the measurement accurate enough that a high probability of occupancy is assigned at the measurement, with most of the area in front of it assigned the minimum. The transition from empty to unknown becomes more gradual with measurements further away, until with $z_t = 50$ metres nearly no increased probability above 0.5 are assigned around the measurement.



(a) (b)

Figure 5.6: Responses from Andert's ISM with varying measurement ranges. In (a) certainty around the measurement decreases at more distant ranges, and the transition from empty to unknown becomes more gradual. (b) shows more distant measurements. These models resemble the ideal model with a slow exponential transition toward the measured distance.

Parameter 2: Pixel Noise Standard Deviation (σ_d)

The uncertainty in the range measurements is controlled by the standard deviation of the pixel disparity σ_d. This model reacts very similarly to the model by Thrun to increased pixel noise, as shown in Figure 5.7.

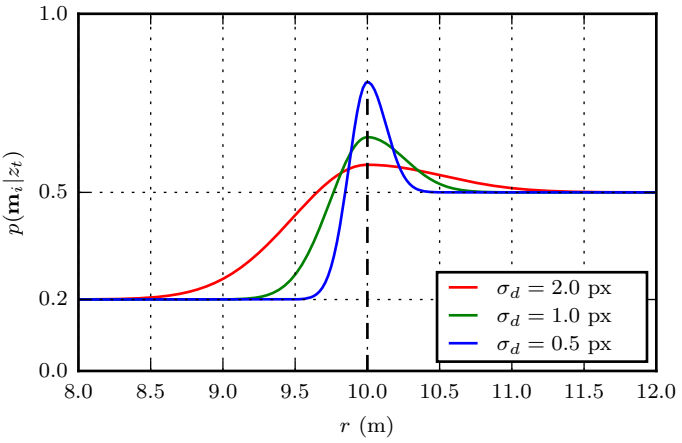


Figure 5.7: A number of different models using Andert's inverse sensor model and various amounts of pixel error.

As the pixel disparity noise increases, the transition from p_{min} to unknown becomes more gradual, and the maximum calculated occupancy probability decreases. The result of this is that the occupancy probabilities of cells between the sensor and the measurement will be decreased less, and at the measurement increased less.

Parameter 3: Measurement Significance (k)

The significance constant k controls how much impact a single measurement has on the map. As shown in Figure 5.8, a higher value of k decreases the impact of the exponential part of the model from Equation 5.10.

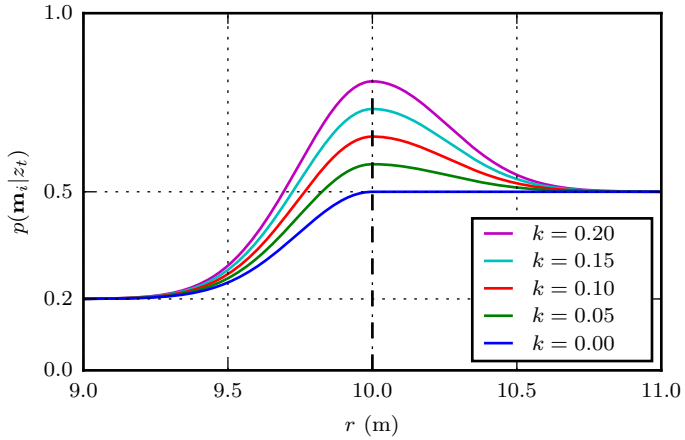


Figure 5.8: Different sensor models for the same measurement, with varying measurement significances (k). Higher values of k mean that blocks are more likely to be considered occupied.

A higher value of k will increase the number of cells whose occupancy probabilities will be raised. This means that the sensor model can be customised slightly for a particular application, as this constant can be chosen freely.

5.1.2.3 Conclusion

The inverse sensor model as derived by Andert offers a good solution for mapping, since it not only performs as expected for near measurements but also handles far measurements acceptably. Since it tends to a constant 0.5 for very distant measurements, it seems to be a safer model to use for mapping than Thrun's. It tends to this limit very slowly, however, and for most measurement distances the occupancy probability will be reduced for any map cell between the sensor and the measurement origin. Regardless, the model is not the result of a sound derivation and is simply stated. It also only supports a very simple noise model, which may be insufficient for stereo vision sensors with more complex error characteristics.

5.2 Principled Inverse Sensor Model

The field of autonomous mapping may benefit from a more principled derivation for the inverse sensor model used in the occupancy grid mapping algorithm.

After stating the objectives, we provide such a derivation here for a stereo vision sensor, and expand it to be compatible with any realistic sensor error distribution. After this, we discuss the assumptions we base the model on, the implementation of the system, and the parameters of the resulting model. Lastly, we draw some conclusions about the new inverse sensor model.

5.2.1 Objectives

As shown in Section 5.1, there is a need for a more principled derivation of the inverse sensor model for stereo vision. Instead of a model that employs mathematical operations intuitively to get the desired result, there is a need for a derivation that is done from sound statistical principles and under clear and meaningful assumptions.

Furthermore, a more robust model that can handle a wide variety of measurements will be more powerful than the currently available methods. We showed that the existing models are not adept at handling very distant measurements, due to the measurement noise of stereo vision being distance dependent. It should be able to handle inaccurate measurements and not let them negatively influence the map. A conservative approach is preferred, that will rather leave the map unaffected than let bad measurements decrease its reliability.

Another shortcoming in existing systems is that they do not allow for a more complex noise distribution than a unimodal Gaussian. Although this is an adequate noise model for many sparse sensors, more complete sensors such as dense stereo often result in more complex error characteristics (see Section 3.3).

5.2.2 Derivation

The basic derivation of the principled inverse sensor model (PRISM) as provided below was done with assistance from Dr CE van Daalen. The derivation from this point to the end of Section 5.2.2.4 is based on work done by him, but has not been published before in any form. It offers a configurable model that can be applied to any sensor, but in this application the focus is on dense stereo as input.

The 1-dimensional measurement ray approximation is made, as discussed in Section 4.4.1, under the assumption that accurate robot state information is available and that the map cells are at an appropriate resolution.

For mathematical simplicity, the measurement ray is split into three sections by the map cell currently being evaluated (cell i), as shown in Figure 5.9. The section of measurement ray from the sensor to the closest edge of cell i is denoted by A, within the cell is called B, and beyond it C. The distances along the measurement ray describing the limits of the three sections are $r_{\min}$, r_L, r_H, and $r_{\max}$ – where $r_{\min}$ and $r_{\max}$ are the closest and furthest possible measurements that the sensor could make, and r_L and r_H are the limits of cell i along the measurement ray.

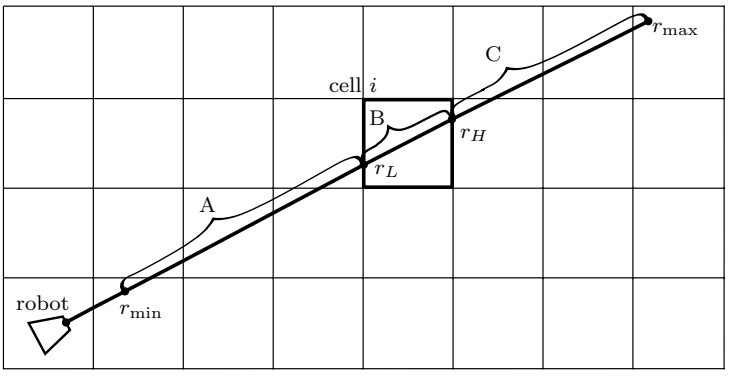


Figure 5.9: Measurement ray split into three sections (A, B, and C), with the cell currently being evaluated (cell i) marked.

These regions are used to combine the cells along the measurement ray. Combining the cells in this way simplifies the derivation, but does not affect its accuracy because no approximations or assumptions

are made. For every cell being evaluated, the distribution over the part in region A being occupied or empty is m_A, in region B the distribution is still defined as m_i, and in C is assigned m_C. The important distinction to make when updating a map cell is whether the measurement originated between the sensor and it, within it, or further than it. Therefore, these combinations of cells describe the map along the measurement ray sufficiently to update cell i, as will become clear later in the derivation. It is also important to note that the dividing of the map into cells is arbitrary, since we model the underlying environment and not the map.

As before, we wish to calculate the inverse sensor model in Equation 5.1, which can be restated using Bayes' theorem as

$$p(\mathbf{m}_i|z_t) = \frac{p(z_t|\mathbf{m}_i)p(\mathbf{m}_i)}{p(z_t)}. \tag{5.13}$$

This is marginalised over all the cells in the map except for m_i ($m \setminus m_i$) such that

$$\begin{aligned} p(m_i|z_t) &= \sum_{m\setminus m_i} \frac{p(z_t|m)p(m)}{p(z_t)} \\ &= \eta \sum_{m\setminus m_i} p(z_t|m)p(m), \end{aligned} \tag{5.14}$$

where η is a normalising constant so that $p(m_i|z_t)$ is a valid distribution. The binary distribution of cell i ($p(m_i)$) can be related to the probability that it is occupied ($p(\mathbf{m}_i|z_t)$) by assigning a value of occupied to cell i.

Although it can be very computationally expensive to sum over all the map cells other than cell i, only the cells along the measurement ray need to be considered. Since we assumed that the measurement beam can be approximated by a thin ray, only the cells along it can impact the measurement. Hence, a measurement only provides information about the set of cells along the path of the ray that approximates it.

Equation 5.14 requires solutions for the map prior $p(m)$, as well as the measurement distribution $p(z_t|m)$. This measurement distribution can be further expanded by marginalising over all possible values for the actual range to the nearest object r, and using conditional probability:

$$\begin{aligned} p(z_t|m) &= \int_{-\infty}^{\infty} p(z_t, r|m)dr \\ &= \int_{-\infty}^{\infty} p(z_t|r, m)\ p(r|m)dr \\ &= \int_{-\infty}^{\infty} p(z_t|r)\ p(r|m)dr. \end{aligned} \tag{5.15}$$

An assumption made in the last step of Equation 5.15 is that the probability of a measurement is conditionally independent of the map (m) given the true range to the closest obstacle (r). This is a realistic assumption, since the map provides no additional information about the measurement when the true range is available. The distribution $p(z_t|r)$ provides the probabilities that distance measurements would be made, which is conditionally independent of the map if we know how far the nearest object is along the measurement ray.

Using this, the full function for the sensor model can be written at this stage as

$$p(m_i|z_t) = \eta \sum_{m \setminus m_i} \left(\int_{-\infty}^{\infty} p(z_t|r)\ p(r|m) dr \right) p(m), \tag{5.16}$$

which shows the required distributions to be the sensor distribution $p(z_t|r)$ and the true range distribution $p(r|m)$.

5.2.2.1 Sensor Distribution

The sensor distribution $p(z_t|r)$ is the distribution of a measurement given the range to the nearest obstacle. It describes how accurate a sensor is, since an accurate sensor will be likely to create measurements close to the nearest obstacle.

We use a stereo vision sensor that measures a correct disparity value d, which can be converted to a distance r using the approximation

$$r = \frac{fb}{d}. \tag{5.17}$$

This pixel measurement is preliminarily assumed to be made with a measurement error described by a Gaussian distribution of the form

$$n \sim \mathcal{N}(0,\ {\sigma_d}^2), \tag{5.18}$$

with σ_d the standard deviation of the disparity error. The probability density function is the common Gaussian model

$$p(n) = \frac{1}{\sqrt{2\pi {\sigma_d}^2}}\ \exp\left(-\frac{n^2}{2{\sigma_d}^2}\right). \tag{5.19}$$

The distance measurement z_t is converted from the noisy disparity measurement (similar to Equation 5.17) using

$$z_t = \frac{fb}{d+n} = T(n), \tag{5.20}$$

where f and b are the focal length and baseline respectively of the stereo sensor. This is combined with Equation 5.18 to get the inverse

$$n = T^{-1}(z_t) = \frac{fb}{z_t} - \frac{fb}{r} = fb\left(\frac{1}{z_t} - \frac{1}{r}\right). \tag{5.21}$$

This inverse transform to the random variable n is monotonically increasing. Therefore, we can get

the transformed probability density function from the distribution of the measurement [64] as follows:

$$\begin{aligned} p(z_t|r) &= p(n)\left|\frac{dn}{dz_t}\right| \\ &= p\left(T^{-1}(z_t)\right)\left|\frac{dT^{-1}(z_t)}{dz_t}\right| \\ &= \frac{1}{\sqrt{2\pi\sigma_d{}^2}}\ \frac{fb}{z_t^2}\ \exp\left(-\frac{n^2}{2\sigma_d^2}\ \right) \\ &= \frac{fb}{z_t^2\sqrt{2\pi\sigma_d{}^2}}\ \exp\left(-\frac{1}{2\sigma_d}\left(f^2b^2\left(\frac{1}{z_t}-\frac{1}{r}\right)^2\right)\right) \\ &= \frac{fb}{z_t^2\sqrt{2\pi\sigma_d{}^2}}\ \exp\left(-\frac{1}{2}\left(\frac{fb}{\sigma_d}\right)^2\left(\frac{1}{z_t}-\frac{1}{r}\right)^2\right). \end{aligned} \tag{5.22}$$

This provides a distribution that a measurement is caused by an obstacle at a certain distance, and requires only an error distribution for the disparity measurements, and a way to convert the disparity measurements to distance measurements.

5.2.2.2 True Range Distribution

Next we consider the true range distribution in Equation 5.15, which describes the probabilities for different distances to the closest obstacle given information about the map ($p(r|m)$). This is done in the context of the measurement ray as defined in Figure 5.9.

Since there is no information available initially about the obstacles in the environment, we start with the assumption that objects are randomly distributed in the world. Additionally, the locations of different obstacles can be assumed to be statistically independent, and hence encountering an obstacle along the measurement ray can be considered a Poisson process with a rate of α.

From this it follows that the probability density function of the distance to the first obstacle will be exponential [65] with the same rate α. The exponential nature of this function implies that in an environment with randomly distributed obstacles it is most likely that the first obstacle is near, becoming less likely with increased distance. The density of objects in the environment controls how quickly this function decreases, and in turn how close the first obstacle is expected to be.

This leads to a probability density function $p(r)$ as shown in Figure 5.10(a). Without any other information, the probability density function can be written as $p(r) = u(r) \times \alpha \exp(-\alpha r)$ where $u(r)$ is the unit step function. A high value of α leads to a fast decay, causing a high probability of the first obstacle being near the sensor.

This distribution can be updated if the distance to the nearest obstacle is known to be within certain boundaries. Conditioning the true range distribution in the range r_1 to r_2 leads to

$$\begin{aligned} p(r|r_1 < r \leq r_2) &= \frac{\alpha\exp(-\alpha r)\left(u(r-r_2)-u(r-r_1)\right)}{\int_{r_1}^{r_2}\alpha\exp(-\alpha r)\,dr} \\ &= \frac{\alpha\exp(-\alpha r)\left(u(r-r_2)-u(r-r_1)\right)}{\exp(-\alpha r_2)-\exp(-\alpha r_1)}, \end{aligned} \tag{5.23}$$

which can be seen in Figure 5.10(b).

Finally, the conditional true range distribution is calculated for every combination of occupation for the three sections (A, B, and C) into which we split the measurement ray in Figure 5.9. This leads to the eight combinations of occupancy shown in Table 5.2. Each scenario is detailed with the three regions'

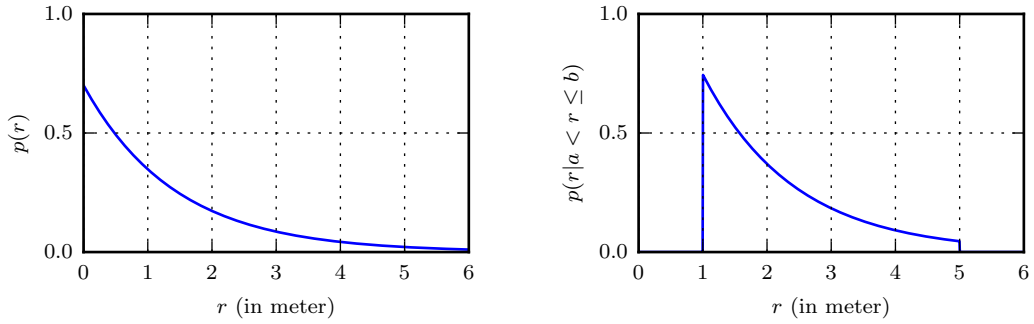


Figure 5.10: (a) The probability of the first obstacle occurring at a certain distance, for an α of 1 and no information about the map. (b) The updated probability density function given boundaries for the first obstacle of $r_1 = 1$ m and $r_2 = 5$ m.

occupancies in the first three columns, followed by the first region along the beam that is occupied. Lastly, the true range distribution is shown in the last column as calculated using Equation 5.23.

m_A	m_i	m_C	First occupied region	$p(r\|m)$
0	0	0	—	$\frac{\alpha \exp(-\alpha r)}{\exp(-\alpha r_{\max}) - \exp(-\alpha r_H)}$
0	0	1	C	$\frac{\alpha \exp(-\alpha r)}{\exp(-\alpha r_H) - \exp(-\alpha r_L)}$
0	1	0	B	$\frac{\alpha \exp(-\alpha r)}{\exp(-\alpha r_H) - \exp(-\alpha r_L)}$
0	1	1		
1	0	0	A	$\frac{\alpha \exp(-\alpha r)}{\exp(-\alpha r_L) - \exp(-\alpha r_{\min})}$
1	0	1		
1	1	0		
1	1	1		

Table 5.2: The conditioned distribution of the distance to the closest obstacle for each possible scenario. The sections A, B, C are used as described in Figure 5.9.

This calculates the true range distribution based on which is the first occupied region along the measurement ray.

5.2.2.3 Map Prior

The prior probability of a map cell is its probability of occupancy given no information other than its dimensions. In conventional implementations this is usually chosen to be 0.5, but instead here the implications of varying the map prior are investigated. The required value is the full map prior $p(m)$ as used in Equation 5.16, which is the distribution describing how the entire map is occupied before any information is provided.

Since the state of a map cell is defined as being either occupied or empty, its prior probability can be described by a binary distribution. The two discrete values in this function represent the prior probability

of the map cell to be empty (at 0) and of being occupied (at 1).

Using the same process as in Section 5.2.2.2, a map cell's prior can be calculated as a function of its range boundaries on the measurement ray. As an example, cell i is bound by r_L and r_H and therefore has a prior described by

$$\begin{aligned} p(\mathbf{m}_i) &= 1 - \exp\left(-\alpha(r_H - r_L)\right) \text{ and} \\ p(\overline{\mathbf{m}}_i) &= \exp\left(-\alpha(r_H - r_L)\right), \end{aligned} \tag{5.24}$$

using the fact that the two probabilities must sum to 1.

To get a chosen prior probability $p(\mathbf{m}_i)$, the density value α is chosen using

$$\begin{aligned} p(\mathbf{m}_i) &= 1 - \exp\left(-\alpha(r_H - r_L)\right) \\ \alpha &= \frac{\log(1 - p(\mathbf{m}_i))}{r_L - r_H}. \end{aligned} \tag{5.25}$$

Now a full prior probability for the map $p(m)$ can be calculated for each possible combination of occupancy for the regions described by A, B, and C. The three cells are assumed to be statistically independent according to the definition of the occupancy grid map (see Section 4.3), and therefore the full prior distribution can be written as

$$p(m) = p(m_A, m_i, m_C) = p(m_A)p(m_i)p(m_C). \tag{5.26}$$

All of the possible combinations of the three occupancies are shown in Table 5.3, along with the full map prior of each. Adjacent empty cells have been combined for simplicity, since $(\exp(A) \times \exp(B)) = \exp(A + B)$.

m_A	m_i	m_C	$p(m)$
0	0	0	$\exp\left(-\alpha(r_{\max} - r_{\min})\right)$
0	0	1	$\exp\left(-\alpha(r_H - r_{\min})\right)\left(1 - \exp\left(-\alpha(r_{\max} - r_H)\right)\right)$
0	1	0	$\exp\left(-\alpha(r_L - r_{\min})\right)\left(1 - \exp\left(-\alpha(r_H - r_L)\right)\right)\exp\left(-\alpha(r_{\max} - r_H)\right)$
0	1	1	$\exp\left(-\alpha(r_L - r_{\min})\right)\left(1 - \exp\left(-\alpha(r_H - r_L)\right)\right)\left(1 - \exp\left(-\alpha(r_{\max} - r_H)\right)\right)$
1	0	0	$\left(1 - \exp\left(-\alpha(r_L - r_{\min})\right)\right)\exp\left(-\alpha(r_{\max} - r_L)\right)$
1	0	1	$\left(1 - \exp\left(-\alpha(r_L - r_{\min})\right)\right)\exp\left(-\alpha(r_H - r_L)\right)\left(1 - \exp\left(-\alpha(r_{\max} - r_H)\right)\right)$
1	1	0	$\left(1 - \exp\left(-\alpha(r_L - r_{\min})\right)\right)\left(1 - \exp\left(-\alpha(r_H - r_L)\right)\right)\exp\left(-\alpha(r_{\max} - r_H)\right)$
1	1	1	$\left(1 - \exp\left(-\alpha(r_L - r_{\min})\right)\right)\left(1 - \exp\left(-\alpha(r_H - r_L)\right)\right)\left(1 - \exp\left(-\alpha(r_{\max} - r_H)\right)\right)$

Table 5.3: Full map prior probability $p(m)$ for each possible combination of occupancy for the combined cells described by the segments A to C.

This enables the calculation of the a priori map probability based only on the size of a map cell and the rate of the Poisson process describing the event of obstacles occurring along the measurement ray.

5.2.2.4 Calculation of Model

All that remains at this point is to combine all the functions and simplify it to a point where it can be computed directly.

We evaluate Equation 5.14, which describes the distribution for the occupancy of cell i given a

measurement z_t, for the case where cell i is occupied. The distribution $p(z_t|m)$ is described given an interval for the true range r, which represents the region in which the true range is (one of either m_A, m_i, or m_C). Combining the true range interval with the different possible combinations of the map from Table 5.3, it can provide all the required information about the map.

The probability of cell i being occupied given z_t is now written as

$$\begin{aligned}
&p(\mathbf{m}_i|z_t) \\
&= \eta \left[p(z_t|r_L < r \leq r_H) \left(p(\overline{\mathbf{m}}_A, \mathbf{m}_i, \overline{\mathbf{m}}_C) + p(\overline{\mathbf{m}}_A, \mathbf{m}_i, \mathbf{m}_C)\right)\right. \\
&\quad \left. + p(z_t|r_{\min} < r \leq r_L) \left(p(\mathbf{m}_A, \mathbf{m}_i, \overline{\mathbf{m}}_C + p(\mathbf{m}_A, \mathbf{m}_i, \mathbf{m}_C)\right)\right] \\
&= \eta \left[p(z_t|r_L < r \leq r_H) p(\overline{\mathbf{m}}_A, \mathbf{m}_i) + p(z_t|r_{\min} < r \leq r_L) p(\mathbf{m}_A, \mathbf{m}_i)\right] \\
&= \eta \left[p(z_t|r_L < r \leq r_H) \left(\exp\left(-\alpha(r_L - r_{\min})\right)\right) \left(1 - \exp\left(-\alpha(r_H - r_L)\right)\right)\right. \\
&\quad \left. + p(z_t|r_{\min} < r \leq r_L) \left(1 - \exp\left(-\alpha(r_L - r_{\min})\right)\right) \left(1 - \exp\left(-\alpha(r_H - r_L)\right)\right)\right].
\end{aligned} \tag{5.27}$$

A similar process is followed for cell i being empty:

$$\begin{aligned}
&p(\overline{\mathbf{m}}_i|z_t) \\
&= \eta \left[p(z_t|r_H < r \leq r_{\max}) p(\overline{\mathbf{m}}_A, \overline{\mathbf{m}}_i, \mathbf{m}_C)\right. \\
&\quad \left. + p(z_t|r_{\min} < r \leq r_L) \left(p(\mathbf{m}_A, \overline{\mathbf{m}}_i, \overline{\mathbf{m}}_C + p(\mathbf{m}_A, \overline{\mathbf{m}}_i, \mathbf{m}_C)\right)\right] \\
&= \eta \left[p(z_t|r_H < r \leq r_{\max}) p(\overline{\mathbf{m}}_A, \overline{\mathbf{m}}_i, \mathbf{m}_C) + p(z_t|r_{\min} < r \leq r_L) p(\mathbf{m}_A, \overline{\mathbf{m}}_i)\right] \\
&= \eta \left[p(z_t|r_H < r \leq r_{\max}) \left(\exp\left(-\alpha(r_H - r_{\min})\right)\right) \left(1 - \exp\left(-\alpha(r_{\max} - r_H)\right)\right)\right. \\
&\quad \left. + p(z_t|r_{\min} < r \leq r_L) \left(1 - \exp\left(-\alpha(r_L - r_{\min})\right)\right) \exp\left(-\alpha(r_H - r_L)\right)\right].
\end{aligned} \tag{5.28}$$

Note that the case where all three segments are empty is ignored since it cannot generate a measurement.

Next, from Equation 5.27 we expand the distribution of the measurement z_t given an arbitrary interval for the general true range $r_1 < r \leq r_2$, by using the sensor distribution from Equation 5.22 and the true range distribution from Table 5.2. This yields

$$\begin{aligned}
&p(z_t|r_1 < r \leq r_2) \\
&= \int_{-\infty}^{\infty} p(z_t|r) p(r|r_1 < r \leq r_2) dr \\
&= \int_{r_1}^{r_2} \frac{fb}{z_t^2 \sqrt{2\pi\sigma_d^2}} \exp\left(-\frac{1}{2}\left(\frac{fb}{\sigma_d}\right)^2 \left(\frac{1}{z_t} - \frac{1}{r}\right)^2\right) \left(\frac{\alpha \exp(-\alpha r)}{\exp(-\alpha r_2) - \exp(-\alpha r_1)}\right) dr \\
&= \frac{\alpha}{z_t^2 (\exp(-\alpha r_1) - \exp(-\alpha r_2))} \int_{r_1}^{r_2} \frac{1}{\sqrt{2\pi \left(\frac{\sigma_d}{fb}\right)^2}} \exp\left(-\frac{1}{2}\left(\frac{fb}{\sigma_d}\right)^2 \left(\frac{1}{z_t} - \frac{1}{r}\right)^2 - \alpha r\right) dr \\
&= \frac{\alpha}{z_t^2 (\exp(-\alpha r_1) - \exp(-\alpha r_2))} I_{r_1}^{r_2}(z_t),
\end{aligned} \tag{5.29}$$

where $I_{r_1}^{r_2}(z_t)$ is used only to simplify notation.

When this is substituted into Equation 5.27, the following simplifications can be made:

$$
\begin{aligned}
&p(\mathbf{m}_i|z_t) \\
&\quad = \eta \left[\frac{\alpha}{z_t^2(\exp(-\alpha r_L) - \exp(-\alpha r_H))} \boldsymbol{I}_{r_L}^{r_H}(z_t) \left(\exp(-\alpha(r_L - r_{\min}))\right)\left(1 - \exp(-\alpha(r_H - r_L))\right) \right. \\
&\qquad \left. + \frac{\alpha}{z_t^2(\exp(-\alpha r_{\min}) - \exp(-\alpha r_L))} \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \left(1 - \exp(-\alpha(r_L - r_{\min}))\right)\left(1 - \exp(-\alpha(r_H - r_L))\right) \right] \\
&\quad = \frac{\eta\alpha}{z_t^2} \left[\frac{\exp(-\alpha(r_L - r_{\min}))\left(1 - \exp(-\alpha(r_H - r_L))\right)}{\exp(-\alpha r_L) - \exp(-\alpha r_H)} \boldsymbol{I}_{r_L}^{r_H}(z_t) \right. \\
&\qquad \left. + \frac{\left(1 - \exp(-\alpha(r_L - r_{\min}))\right)\left(1 - \exp(-\alpha(r_H - r_L))\right)}{\exp(-\alpha r_{\min}) - \exp(-\alpha r_L)} \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \right] \\
&\quad = \frac{\eta\alpha}{z_t^2} \left[\exp(\alpha r_{\min}) \boldsymbol{I}_{r_L}^{r_H}(z_t) + \exp(\alpha r_{\min}) \left(1 - \exp(-\alpha(r_H - r_L))\right) \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \right] \\
&\quad = \eta\alpha \exp(\alpha r_{\min}) z_t^{-2} \left[\boldsymbol{I}_{r_L}^{r_H}(z_t) + \left(1 - \exp(-\alpha(r_H - r_L))\right) \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \right]
\end{aligned}
\tag{5.30}
$$

and similarly for the complement:

$$
\begin{aligned}
&p(\overline{\mathbf{m}}_i|z_t) \\
&\quad = \eta \left[\frac{\alpha}{z_t^2(\exp(-\alpha r_H) - \exp(-\alpha r_{\max}))} \boldsymbol{I}_{r_H}^{r_{\max}}(z_t) \left(\exp(-\alpha(r_H - r_{\min}))\right)\left(1 - \exp(-\alpha(r_{\max} - r_H))\right) \right. \\
&\qquad \left. + \frac{\alpha}{z_t^2(\exp(-\alpha r_{\min}) - \exp(-\alpha r_L))} \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \left(1 - \exp(-\alpha(r_L - r_{\min}))\right)\exp(-\alpha(r_H - r_L)) \right] \\
&\quad = \frac{\eta\alpha}{z_t^2} \left[\frac{\left(\exp(-\alpha(r_H - r_{\min}))\right)\left(1 - \exp(-\alpha(r_{\max} - r_H))\right)}{\exp(-\alpha r_H) - \exp(-\alpha r_{\max})} \boldsymbol{I}_{r_H}^{r_{\max}}(z_t) \right. \\
&\qquad \left. + \frac{\left(1 - \exp(-\alpha(r_L - r_{\min}))\right)\exp(-\alpha(r_H - r_L))}{\exp(-\alpha r_{\min}) - \exp(-\alpha r_L)} \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \right] \\
&\quad = \frac{\eta\alpha}{z_t^2} \left[\exp(\alpha r_{\min}) \boldsymbol{I}_{r_H}^{r_{\max}}(z_t) + \exp(\alpha r_{\min}) \exp(-\alpha(r_H - r_L)) \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \right] \\
&\quad = \eta\alpha \exp(\alpha r_{\min}) z_t^{-2} \left[\boldsymbol{I}_{r_H}^{r_{\max}}(z_t) + \exp(-\alpha(r_H - r_L)) \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \right].
\end{aligned}
\tag{5.31}
$$

It only remains to calculate the occupancy ratio as defined in Section 4.2:

$$
\begin{aligned}
&\frac{p(\mathbf{m}_i|z_t)}{p(\overline{\mathbf{m}}_i|z_t)} \\
&\qquad = \frac{\eta\alpha \exp(\alpha r_{\min}) z_t^{-2} \left[\boldsymbol{I}_{r_L}^{r_H}(z_t) + \left(1 - \exp(-\alpha(r_H - r_L))\right) \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \right]}{\eta\alpha \exp(\alpha r_{\min}) z_t^{-2} \left[\boldsymbol{I}_{r_H}^{r_{\max}}(z_t) + \exp(-\alpha(r_H - r_L)) \boldsymbol{I}_{r_{\min}}^{r_L}(z_t) \right]} \\
&\qquad = \frac{\boldsymbol{I}_{r_L}^{r_H}(z_t) + \left(1 - \exp(-\alpha(r_H - r_L))\right) \boldsymbol{I}_{r_{\min}}^{r_L}(z_t)}{\boldsymbol{I}_{r_H}^{r_{\max}}(z_t) + \exp(-\alpha(r_H - r_L)) \boldsymbol{I}_{r_{\min}}^{r_L}(z_t)},
\end{aligned}
\tag{5.32}
$$

which can be used to retrieve the occupancy probability.

This concludes the derivation of our inverse sensor model using a unimodal Gaussian distribution to describe the sensor error. The occupancy ratio shown in Equation 5.32 can be implemented directly into Equation 4.7 to update the occupancy probability of a map cell with a new measurement.

5.2.3 Generalising the Sensor Distribution

At this stage a zero-mean Gaussian distribution can be used to describe the sensor noise. However, the pixel measurements from a stereo algorithm may be offset with some constant value, possibly due to incorrect matching or a problem with how the cameras are set up. We allow for this by extending the sensor model to a general Gaussian distribution, changing the noise model from Equation 5.18 to

$$n \sim \mathcal{N}(\mu_d,\ {\sigma_d}^2) \tag{5.33}$$

where μ_d is the offset or mean of the pixel error distribution.

This alters the distribution of the noise from Equation 5.19 to

$$p(n) = \frac{1}{\sqrt{2\pi{\sigma_d}^2}} \exp\left(\frac{-(n-\mu_d)^2}{2{\sigma_d}^2}\right) \tag{5.34}$$

which changes Equation 5.22 to

$$\begin{aligned} p(z_t|r) &= \frac{1}{\sqrt{2\pi{\sigma_d}^2}}\ \frac{fb}{z_t^2}\ \exp\left(-\frac{(n-\mu_d)^2}{2{\sigma_d}^2}\right) \\ &= \frac{fb}{z_t^2\sqrt{2\pi{\sigma_d}^2}}\ \exp\left(-\frac{1}{2\sigma_d}\left(f^2b^2\left(\frac{1}{z_t}-\frac{1}{r}-\frac{\mu_d}{fb}\right)^2\right)\right) \\ &= \frac{fb}{z_t^2\sqrt{2\pi{\sigma_d}^2}}\ \exp\left(-\frac{1}{2}\left(\frac{fb}{\sigma_d}\right)^2\left(\frac{1}{z_t}-\frac{1}{r}-\frac{\mu_d}{fb}\right)^2\right). \end{aligned} \tag{5.35}$$

This extends the sensor distribution to have an arbitrary mean μ_d. To allow the system to handle a sensor with any arbitrary noise characteristics, a more complex distribution for the error is required. Gaussian mixture models (GMMs), as developed by Feller [66], offer a good solution, since they are capable of approximating any realistic distribution. The errors from dense stereo vision reconstruction systems have been shown in Section 3.3 to be representable by GMMs.

A GMM is a summed combination of a number of different weighted Gaussian distributions. The general form of a GMM consisting of G components is

$$p(y) = \sum_{i=1}^{G} w_i\ p_i(x), \tag{5.36}$$

with p_i a Gaussian distribution with weight w_i (provided that the weights sum to 1 and $w_i \geq 0$). Since the components are added together, the process of forming the GMM from the different Gaussian components is linear. Therefore, the transformation from Equation 5.21 can be implemented as

$$p(z_t|r) = \sum_{i=1}^{G} w_i p_i\left(T^{-1}(z_t)\right)\left|\frac{dT^{-1}(z_t)}{dz_t}\right|. \tag{5.37}$$

Using this with Equation 5.35, Equation 5.29 can be rewritten to incorporate the GMM:

$$
\begin{aligned}
& p(z_t|r_L < r \leq r_H) \\
&= \int_{-\infty}^{\infty} p(z_t|r)p(r|r_L < r \leq r_H)dr \\
&= \int_{r_L}^{r_H} \left(\sum_{i=1}^{G} \frac{fb}{z_t^2\sqrt{2\pi\sigma_{d_i}^2}} \exp\left(-\frac{1}{2}\left(\frac{fb}{\sigma_{d_i}}\right)^2 \left(\frac{1}{z_t} - \frac{1}{r} - \frac{\mu_{d_i}}{fb}\right)^2 \right) \right) \left(\frac{\alpha\ \exp(-\alpha\ r)}{\exp(-\alpha\ r_H) - \exp(-\alpha\ r_L)} \right) dr \\
&= \frac{\alpha}{z_t^2(\exp(-\alpha r_L) - \exp(-\alpha r_H))} \int_{r_L}^{r_H} \sum_{i=1}^{G} \left\{ \frac{1}{\sqrt{2\pi\left(\frac{\mu_{d_i}}{fb}\right)^2}} \exp\left(-\frac{1}{2}\left(\frac{fb}{\sigma_{d_i}}\right)^2 \left(\frac{1}{z_t} - \frac{1}{r} - \frac{\mu_{d_i}}{fb}\right)^2 - \alpha r \right) \right\} dr \\
&= \frac{\alpha}{z_t^2(\exp(-\alpha r_L) - \exp(-\alpha r_H))} J_{r_L}^{r_H}(z_t),
\end{aligned}
\tag{5.38}
$$

which shows that the changes in the new noise model are contained within the integral ($J_{r_L}^{r_H}(z_t)$). The other equations remain unchanged and can be implemented as is when a Gaussian mixture model is used for the sensor error instead of a unimodal zero-mean Gaussian distribution.

5.2.4 Assumptions

The derivation for the inverse sensor model provided is based on only a few key assumptions, which places minimal limitations and restrictions on the system. All of the assumptions that we made are also connected to logical reasoning based in physical meaning. A summary is provided here.

Firstly, we made some assumptions about the environment. It is assumed that the locations of different obstacles are independent and that they are uniformly distributed with a density α.

Next, it is assumed that any obstacle present in a map cell will cause a measurement. This is based on the map cells being an appropriate size for the actual measurement ray width, and the accuracy of the robot pose estimate. Cases that violate the assumption may cause problems in the calculated map.

Lastly, it is assumed that the measurement distribution is conditionally independent of the map given the true range to the closest obstacle, which is based on the fact that only the nearest object should create a measurement.

5.2.5 Implementation

The calculation of the sensor model is currently very calculation intensive due to the integrals that must be solved. Solving them requires numerical integration, which can take long if the integral is to be approximated accurately.

The integration is done using the trapezoidal technique [67], which offers high accuracy with fairly fast calculation. An integral from a to b can be approximated with

$$
\int_a^b f(x)dx \approx (b-a)\left[\frac{f(a)+f(b)}{2}\right], \tag{5.39}
$$

where it naturally follows that smaller differences between a and b yield a more accurate approximation.

This difference has been chosen empirically as $dx = 1$ mm, which we found to be the largest differential distance to yield an accurate result. To improve execution time in large intervals, the number of integration points are limited to 10000 if the interval is longer than 10 metres. This means that in long integrals the accuracy is reduced and dx increased, but the shorter intervals are calculated with a small dx (i.e. from r_L to r_H).

5.2.6 Parameters

To study the performance of this new inverse sensor model, each of the parameters are varied in turn, similarly as with Thrun's and Andert's models. This offers an indication of the impact of each parameter, and should highlight any problems.

5.2.6.1 Parameter 1: Measurement distance (z_t)

Due to the distance dependent noise inherent to stereo vision, measurements at various distances will have different sensor models. The shape of the distribution changes drastically with increased measurement distance as shown in Figure 5.11.

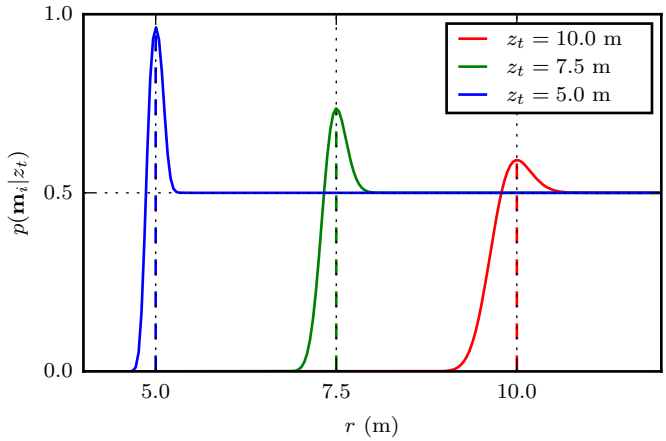


Figure 5.11: Different sensor models created with PRISM for various distances from the sensor. The maximum occupancy probability is shown to decrease with more distant measurements, and the peak is shown to widen.

This increase in measurement distance causes greater uncertainty in the location of the obstacle, leading to a wider peak as well as a decreased maximum occupancy probability.

5.2.6.2 Parameter 2: Pixel Noise Standard Deviation (σ_d)

Increased pixel measurement noise has a similar effect on the shape of the sensor model as increased measurement distance, due to the direct relationship between them. Figure 5.12 shows how increased pixel measurement noise leads to a reduction in the maximum probability of occupancy and a widening of the area of increased occupancy.

5.2.6.3 Parameter 3: Map Cell Size (L)

The size of the map cells impacts the interval that is integrated within cell i in Equation 5.32. We expect that a larger cell size should increase the influence of that part of the function, increasing the maximum

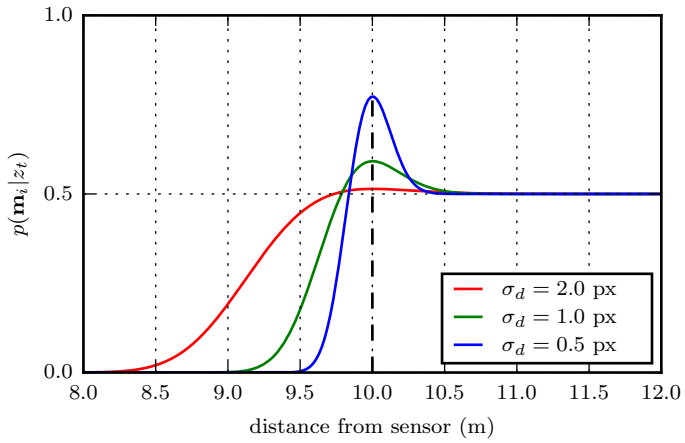


Figure 5.12: Sensor models created with various error standard deviations to show how inaccurate sensors affect the model. It is clear that increased error levels reduce the certainty of occupancy around the measurement location, and stretch the transition from empty to the peak of occupancy probability.

occupancy probability.

The density value α that describes the Poisson process along the measurement ray is chosen for a block size of 0.1 metre, and then the block is varied to visualise its impact. As the block size decreases, the maximum occupancy probability decreases as shown in Figure 5.13. Additionally, for a larger cell the part of the model with an increased occupancy probability enlarges. This is what ensures that the occupancy probability of any map cell within $L/2$ of the maximum of the model will be increased, since it likely contains the object that caused the measurement. We choose the density to get a certain prior probability for a specific cell size, and therefore changing the cell size changes the prior probability that the model tends to for large values of r.

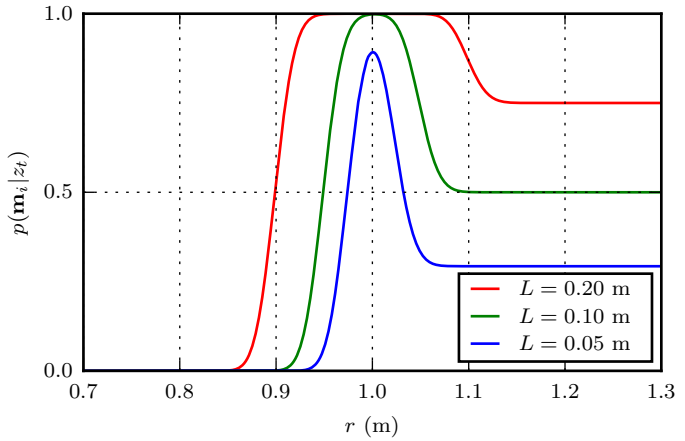


Figure 5.13: Varying cell sizes are used to generate sensor models to show the impact of mapping at different resolutions. A smaller cell leads to reduced occupancy probability.

We see that a smaller cell size will result in a lower prior probability, given a shared density. This ties in with the definition of the occupancy of a map cell in the occupancy grid mapping technique, since for a certain environment the probability that a smaller map cell will contain an obstacle is lower than

that of a larger map cell.

5.2.6.4 Parameter 4: Maximum Influence Distance

One interesting characteristic of our model is that every set of sensor characteristics naturally has a maximum distance at which it can influence the map. This means that over a certain distance, any two measurements would produce practically the same model. They would therefore have the same effect on the map.

This is illustrated in Figure 5.14, with models from measurements at various distances. At first the created models differ substantially, but as the measurement distance increases the models converge. The model with $z_t = 1000$ m is added as reference, and shows that a measurement of any distance has a limited range that it can change the map. Note that the measurement distances of the models are indicated, but only the closest few are within view.

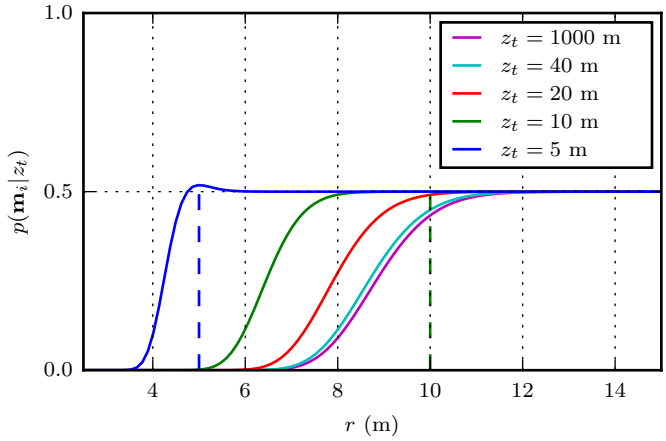


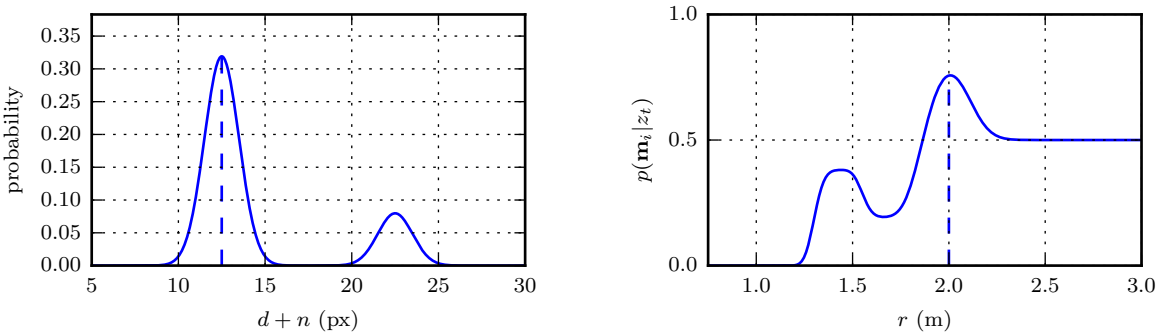
Figure 5.14: Illustration of maximum distance at which our model can influence the map. Sensor used has 1 pixel measurement variance, and the sensor parameters are taken from the KITTI [12] dataset.

This maximum distance is related to the uncertainty in the sensor, and therefore is a combination of the pixel noise, the intrinsic and extrinsic parameters of the stereo sensor, and the cell block size. The relationship between these factors and the maximum distance is rather complex, however, due to the integration and ratios involved in the sensor model. In short, for a high maximum distance of influence the sensor must have a low pixel error, a long focal length, a wide baseline, and a large map cell size.

5.2.6.5 Parameter 5: Number of Components in GMM (G)

A sensor with an error model described by a Gaussian mixture model with multiple components can be modelled by PRISM, as is shown in Figure 5.15. Although this example is exaggerated and such separated components are unlikely to be found with dense stereo sensors, it illustrates that multimodal distributions can be used.

This particular example is a GMM with two components as shown in Figure 5.15(a): one component at the actual range weighted four times greater than the other, which is at an offset of +10 pixels. This offset component indicates a chance that the object may be nearer than the measurement suggests. For simplicity, both these components are modelled with a noise variance of $\sigma_d^2 = 1$ pixel.



(a) (b)

Figure 5.15: (a) multimodal GMM describing sensor noise used to create (b) a sensor model with two components. A component at the true disparity of 12.5 px is weighted 4:1 with one at an offset of 10 pixels.

5.2.7 Conclusion

From the models in Section 5.2.6, PRISM appears to perform well at modelling the uncertainty in measurements. A derivation was shown with minimal assumptions and from sound statistical principles, and therefore the probabilities assigned by our model may be assumed correct under the assumptions made.

Furthermore, it seems to perform well under a variety of parameters, yielding results that make sense. The maximum influence distance is a surprising but logical result, since any practical sensor will have a limited range to which it can reliably gather information. This is especially true for stereo vision, where measured pixel disparities are converted to distances through an inversely proportional function. This maximum influence distance indicates something surprising about stereo vision sensors, however, in that they can only see to a certain distance. It suggests not only that they cannot measure accurately past a certain range, but that an object at that distance cannot be measured. Combining the sensor parameters of a stereo vision sensor and a characterisation of its errors, the furthest possible measurement can be found. Any object that is further than this distance from the sensor will not be visible to the sensor, and any measurement that suggests that it is will be uninformative and will be treated as such by the model.

An important test remains, however, in using our model for mapping an environment. This will be done in Chapter 6, where both 2D and 3D maps are created to establish the usefulness of the model with both synthetic and real world data.

Chapter 6

Results

We test the usefulness and accuracy of our principled inverse sensor model (PRISM) in the context of autonomous mapping by comparing it to some popular inverse sensor models from the literature: the convolution method by Thrun [17], and the method by Andert [47]. In Section 6.1 we note the speeds at which each model incorporates new measurements into a map, in Section 6.2 we discuss and compare the accuracies of the different sensor models, and in Section 6.3 we evaluate maps created from the Tsukuba [21] and KITTI [12] datasets.

6.1 Performance

Our model is designed to calculate the probabilities of occupancy of a map as reliably as possible, which is in contrast to other sensor models that focus at least partly on execution speed. It is still of interest to know how long each model takes to calculate occupancy probabilities, to get an idea of their practical applicability.

All the implementations discussed below are performed on an Intel i5-3570 clocked at 3.4GHz with 12GB RAM. The implementation was done in C++ and for PRISM the GMM was limited to three kernels. All of the models are implemented on the CPU using a single core.

The Octomap implementation [49] we use for creating maps provides a function that calculates the average time taken to incorporate 100,000 measured 3D points. Note that most stereo correspondence algorithms provide between 50,000 and 200,000 data points per time step, and therefore these times are merely an indication of the speed of the sensor models. For consistency, the same set of LIDAR data from the KITTI dataset [12] is used for each of the sensor models. We create a map with each sensor model in turn and the average time for incorporating new values is recorded and shown in Table 6.1. This table also includes a speeded-up version of PRISM, where the integration accuracy has been reduced slightly.

Sensor model	Time per 100,000 data points
Ideal model	0.10 s
Thrun's ISM [17]	0.82 s
Andert's ISM [47]	1.34 s
PRISM	765.69 s
PRISM (speeded-up)	119.19 s

Table 6.1: Performance test of various inverse sensor models.

The other models all calculate occupancy probabilities using closed-form expressions, where PRISM

requires four integrals for each map cell. This leads to an execution time for PRISM orders of magnitude slower than the other models. The speeded-up version does give a substantially reduced execution time, but introduces a slight approximation error.

6.2 Sensor Model Comparison

Since the derivation for PRISM is based only on a few clear assumptions as discussed in Section 5.2.4, and otherwise derived from statistical principles, we may consider its results theoretically correct under the assumptions made. Most other sensor models from literature are designed to calculate a fast approximation. This raises the question of how accurate an approximation each sensor model is.

We aim to answer this by comparing the results of other models to PRISM. Each version of the inverse sensor model is calculated given the sensor characterisation from the KITTI dataset [12]. For the entire range of measurable distances of the sensor, a set of probabilities is calculated using each model. The integral of the squared differences between each model and PRISM is calculated and represents how much that model differs from PRISM. Although this integrated difference value does not have any physical meaning, it can be used to compare the approximations made by different models.

Figure 6.1 shows the results of this comparison. It includes the inverse sensor model from GPSlam [48], which creates models significantly different from those by Thrun and Andert. It is a modification of Andert's ISM, substituting the constant minimum probability for a distance dependent one. The minimum probability it yields tends to 0.5 for distant measurements, which causes the map to not be altered considerably by far measurements that are likely to be inaccurate.

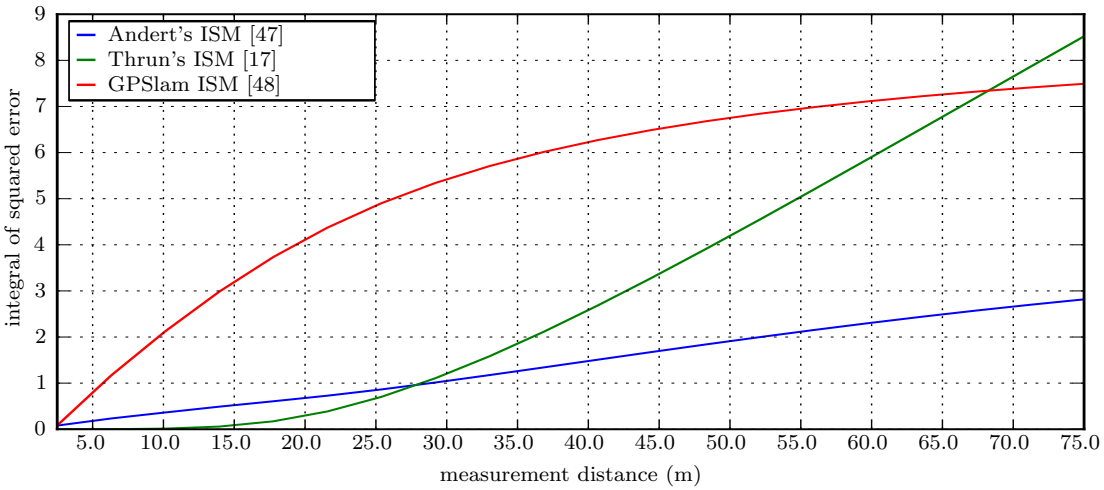


Figure 6.1: Sensor models compared to PRISM, using sensor parameters from the KITTI dataset [12]. The integrated squared differences for each model are calculated for a range of measurements.

The model by Andert [47] has an almost linearly increasing difference with further measurement distances. The convolution method [17] has a considerably faster increasing error over distance, but offers a more accurate approximation for near distances. These increasing differences of both models are because they consider the map empty nearly up to the distance of any measurement, which is in contrast to PRISM that only empties the map to the maximum influence distance.

Contrary to that, the GPSlam ISM [48] tends to a constant 0.5 over the entire range for measurements

at far distances, causing the difference with our model to converge.

As a demonstration of these models when a distant measurement is given, Figure 6.2 shows models created with all four techniques. The differences between the approximations made by the different models have been shown to be greatest at longer distances.

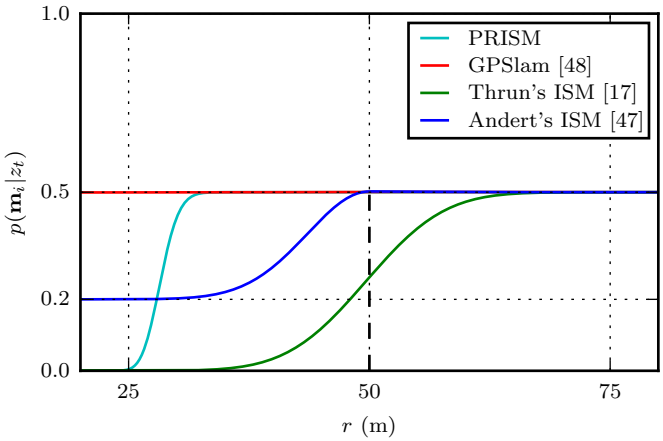


Figure 6.2: A comparison of different ISMs created with a distant measurement at $z_t = 50$ m, demonstrating the approximations made by different models.

6.3 Mapping

To test the performance of PRISM, we use it to create occupancy grid maps. This involves the retrieval of data from either stereo images or from simulation, and triangulating it to world coordinates. Once the data is available each stereo algorithm is characterised as discussed in Section 3.3, and a representative sensor model is created for PRISM, Thrun's [17] and Andert's methods [47].

With the relevant sensor model available, we use the Octomap implementation [49] to create a map from the calculated world coordinates. Octomap calculates a full 3-dimensional map that is difficult to represent in a single image, and therefore a 2-dimensional slice of the map is taken near the height of the sensor. An image of such a slice represents the occupancy grid map, with grey indicating unknown, white empty and black occupied. For illustration some 3D map visualisations are shown at the end of this section.

For a quantitative analysis of the maps a histogram is created of the map errors resulting from each sensor model. These errors are the differences between the created maps from each sensor model and the best possible map, which is created from very accurate LIDAR measurements. A positive difference means that the sensor model overestimates the probability of occupancy, and a negative difference that the model underestimates it. Table 6.2 explains the difference values with some examples. Although this is not the only way to compare two maps, it depicts the distribution of the mapping errors concisely. Other options, such as a confusion matrix, use thresholds to do assignments before comparing two maps and therefore contain much less information.

In this context a false negative is when a cell is incorrectly marked empty, and a false positive is when one is incorrectly marked occupied. All values where the sensor model indicates unknown have been ignored, since this merely indicates unavailable information and not an incorrect assignment. However, values near 0.5 are included.

Map value	Ground truth	Difference	Label
0	0.5	−0.5	False negative
0	1	−1	False negative
1	0	1	False positive
1	0.5	0.5	False positive

Table 6.2: Differences between calculated occupancy probabilities and the ground truth, with labels for each error.

It can be argued that false positives are less problematic for navigation than false negatives, since the navigation algorithm should rather miss a viable path than suggest a path that actually contains an obstacle. On the other hand, in some cases false negatives may cause the robot to miss a viable path and stall. Also note that the presented histograms have been normalised, and therefore indicate the relative frequency between different types of errors.

After some simulated 2D examples, two different 3D datasets are used for testing. The first set is the synthetic dataset from Tsukuba University [21], which offers computer generated stereo images and perfect ground truth range values. The second is the real world dataset from KITTI [12], which is taken using two sophisticated cameras as well as an accurate LIDAR system.

The synthetic set contains closer measurements, since the environment is an indoor office where the furthest walls are only a few metres away. This is combined with a simulated stereo sensor with a short focal length and baseline. This combination makes the sensor accurate for near ranges, which is why we map the environment at a small cell size of 2.5 cm. In the segment of the dataset mapped here the camera does not move much, which means that the area covered is rather small. The real world dataset is mapped at a resolution of 25 cm, since the mapped area is much larger and the sensor is accurate to further distances. The full dataset covers a travel distance of almost 300 metres, which also makes it impractical to map at a finer resolution.

The impact of a smaller resolution on PRISM was discussed in Section 5.2.6.3, and was found to primarily lead to reduced probabilities of occupancy and a shorter maximum influence distance. We note that this may have an effect on mapping the Tsukuba dataset with PRISM.

6.3.1 2D Environment

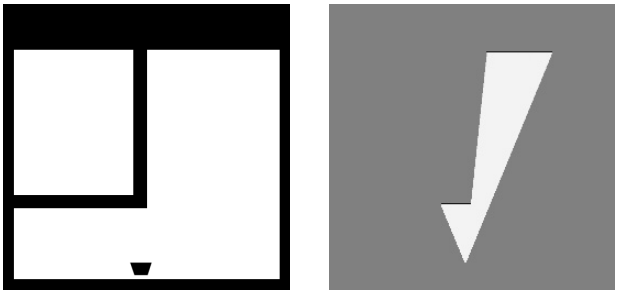
Firstly, some maps are created in 2D of simulated environments. The tests are done by simulating stereo measurements in an environment and using them to create a map.

The results are presented by showing an image of the environment and the ground truth map first, followed by a number of maps generated with different sensor models. Note that the sensor is modelled as a stereo camera system with a field of view of 45°, and various pixel errors are introduced as described.

6.3.1.1 Simple 2D Environment

The first test is shown in Figure 6.3 and features a simple room with one near wall. The ground truth map shows only a single line of map cells marked as occupied at each wall, and the ones between the sensor and these parts as empty.

Maps created with Thrun's ISM and PRISM are shown in Figure 6.4, with the section of the environment visible to the sensor cropped for display purposes. The simulated sensor is characterised to have a 1 pixel measurement error standard deviation but actually measures perfectly, and therefore the two maps are very similar for near obstacles. At the further measurements they differ more in how far



(a) (b)

Figure 6.3: Ground truth for a simple 2D test of a room. (a) the simulated environment with the robot in the bottom centre looking up, and (b) the ground truth map using the ideal sensor model.

from the sensor they empty the map. Note that PRISM only calculates the occupancy probability from a distance $r_{\min}$, which causes the small uncertain area at the bottom of the map.

(a) (b)

Figure 6.4: Maps created using (a) Thrun's ISM and (b) PRISM of the simple 2D room test.

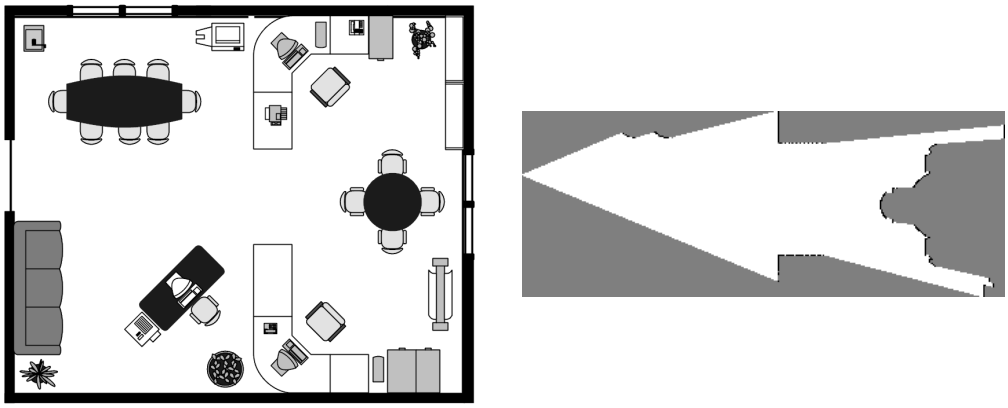
6.3.1.2 2D Office Scene

Next a more complex scene is mapped with measurements containing some error. The image used to simulate this test is shown in Figure 6.5, with a map created using the ideal sensor model and perfect measurements on the right. This ground truth map represents all the information that a sensor with the given sensor parameters could have retrieved from the simulated location.

Noisy measurements are simulated by adding zero-mean Gaussian noise with a 2-pixels standard deviation to the disparity values, with the same data given to each of the sensor models for consistency. The results are shown using four maps: one created with the ideal sensor model to show the effect of the noisy measurements, one with Thrun's ISM, one with Andert's ISM and one with PRISM.

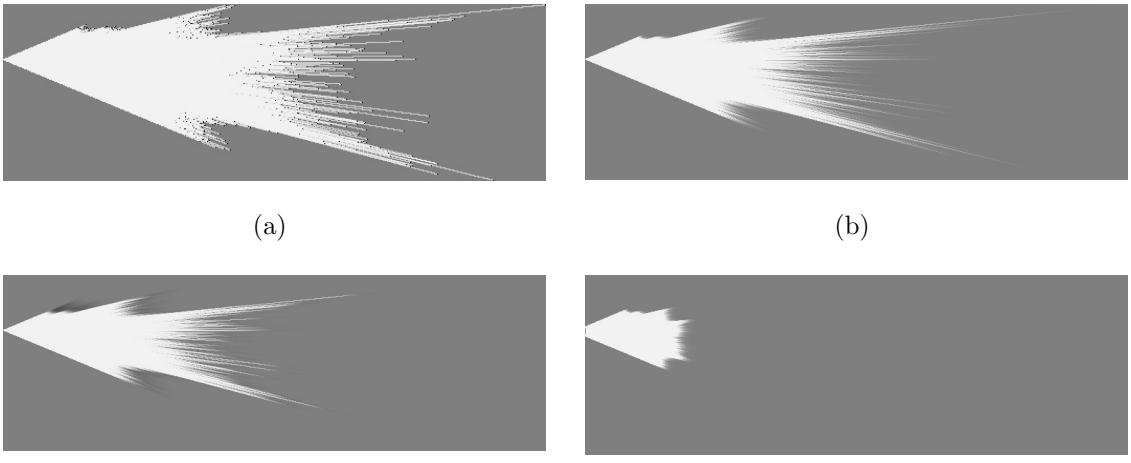
As expected, the measurements are quite inaccurate, as seen in Figure 6.6. With the relatively large sensor error PRISM limits the distance of influence, only emptying a small area near the sensor. This is due to the sensor only providing information to a certain distance, further away than which no measurements can provide information.

The map error histograms are shown in Figure 6.7, and provide a summary of the types of errors caused by each sensor model. They show that Thrun's ISM is more prone to false negatives, and that



(a) (b)

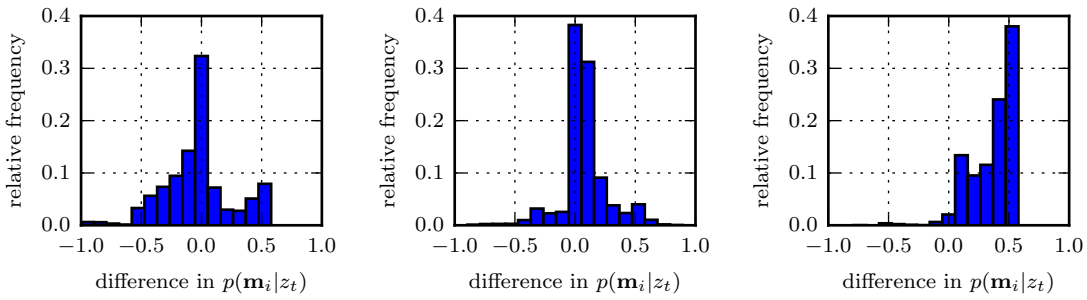
Figure 6.5: Office test ground truth with (a) the simulated environment, where the robot is in the middle left looking right, and (b) the ground truth mapped using the ideal model.



(a) (b)

(c) (d)

Figure 6.6: Office test with zero-mean Gaussian noise. (a) shows the map resulting from the ideal model, and (b) to (d) the maps created using Thrun's ISM, Andert's ISM, and PRISM.



(a) (b) (c)

Figure 6.7: Histograms for the errors made in the mapping of the 2D office simulation with Gaussian noise. (a) uses Thrun's ISM [17], (b) uses Andert's ISM [47], and (c) uses PRISM.

Andert's ISM is slightly more likely to false positives. PRISM seems to be likely to false positives, which occur mostly when the ground truth indicates empty and it indicates somewhat unknown.

This result suggests that Thrun's ISM is the most likely of the three to empty cells incorrectly, and that PRISM mostly overestimates the occupancy probabilities of empty segments by limiting the distance of influence. We expect Andert's method to be more prone to false positives than the other models, especially with increased sensor error, due to its tendency to yield high occupancy probabilities.

6.3.2 Synthetic 3D Dataset

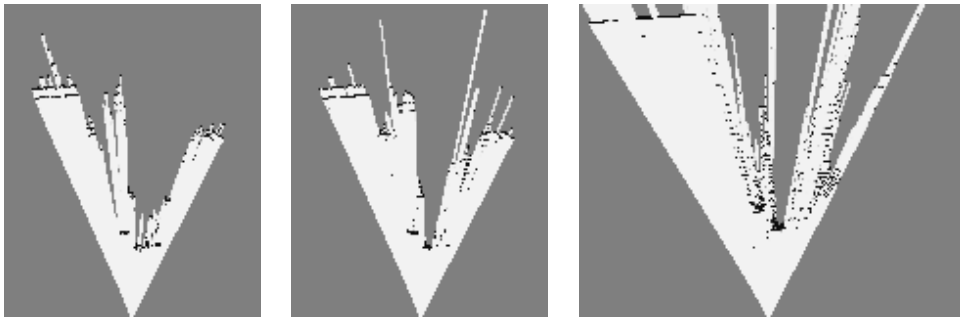
The first test done with full 3D mapping uses the Tsukuba dataset [21], which offers a set of 1800 synthetic stereo image pairs of an indoor office scene. It offers perfect positional information as well as ground truth for the range information. The simulated cameras are set up with a baseline of 10 cm and a focal length of 615 pixels. Some sample images are shown in Figure 6.8. The obstacles marked in the map close to the sensor are the lamp and bust.

(a) (b) (c)

Figure 6.8: Sample images from the Tsukuba dataset [21]. (a) shows an input image, (b) the ground truth disparity map, and (c) the ground truth map.

We characterise each of the stereo algorithms using the provided range ground truth, and create a map at a resolution of 2.5 cm with 50 image pairs. The results are presented with three maps per sensor model: a map created using Thrun's ISM [17], one using Andert's model [47], and one using PRISM. After these maps are shown, we collect the differences between the maps of each sensor model and the ground truth into histograms to evaluate the accuracy of the sensor models.

Maps created using the ideal sensor model and measurements from the three stereo matching algorithms are shown in Figure 6.9, to provide an indication of the accuracy of each technique.



(a) (b) (c)

Figure 6.9: Maps from the Tsukuba dataset using an ideal sensor model, (a) using block matching, (b) semiglobal block matching, and (c) variational matching.

Results for the two block matching based techniques appear very similar, even though the measurements from the semiglobal method are significantly denser. The map from the variational method contains a higher number of incorrect matches that extend past the top of the map shown, and the back of the room appears to be significantly further from the sensor than with the other techniques, suggesting that the results from this correspondence algorithm are less accurate than the other two.

Next maps calculated using the tested models are shown, according to the stereo matching algorithm used. For each algorithm three maps are shown, using Thrun's ISM, Andert's ISM, and PRISM.

6.3.2.1 Block Matching

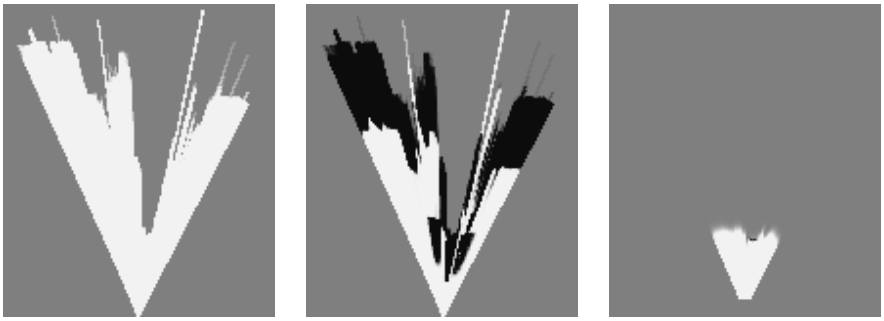
The first algorithm is block matching, and its results are shown in Figure 6.10. The three maps appear to be very different, with Thrun's ISM emptying the map up to every measurement, Andert's method marking a high number of cells as occupied, and PRISM only emptying a short distance into the map. This short distance of PRISM can be attributed to the relatively small block size and the sensor parameters of the simulated stereo sensor. It is important to notice, however, that PRISM is still able to accurately mark the location of the nearest obstacle as occupied.

(a) (b) (c)

Figure 6.10: Maps from Tsukuba dataset using block matching. (a) is created using convolution ISM, (b) Andert's ISM, and (c) PRISM.

6.3.2.2 Semiglobal Block Matching

The next algorithm is semiglobal block matching, of which the results are shown in Figure 6.11. Due to the similarity in measurements, all three maps are similar to the ones drawn of block matching.

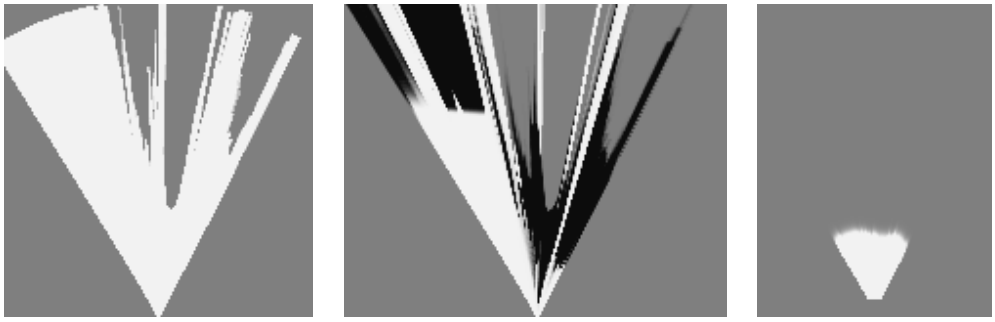


(a) (b) (c)

Figure 6.11: Maps from Tsukuba dataset where semiglobal block matching is used. (a) is created using Thrun's ISM, (b) Andert's ISM, and (c) PRISM.

6.3.2.3 Variational Matching

The variational matching algorithm is suggested by its characterisation to be less accurate for this dataset, with a more complex error distribution. Therefore, the map drawn using PRISM differs greatly from the maps obtained with the other two sensor models, since it limits how far away from the sensor the map can be affected. The increased error also has the effect that the bust and lamp are not marked as occupied.



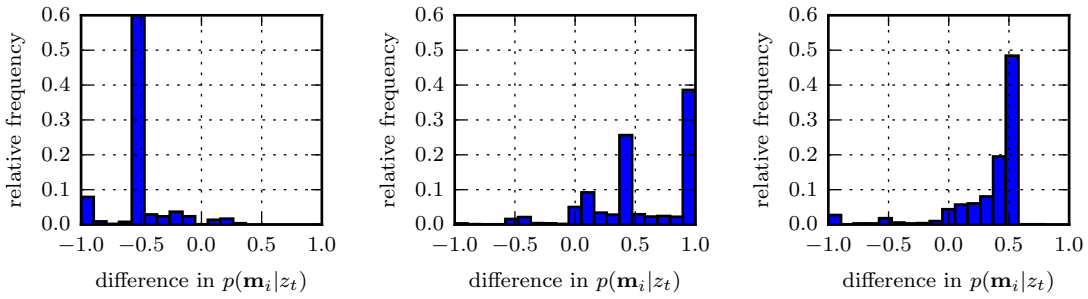
(a) (b) (c)

Figure 6.12: Maps from Tsukuba dataset where variational matching is used. (a) is created using Thrun's ISM, (b) Andert's ISM, and (c) PRISM.

It is interesting to note that although the standard deviation of the pixel error was characterised considerably larger for the variational matching technique than the block matching based ones, its maximum influence distance appears to be similar when using PRISM. This indicates that the other factors (block size and camera parameters) have a stronger impact on the maximum distance.

6.3.2.4 Map Error Histograms

The differences between the maps produced by the different sensor models and the ground truth are presented as histograms in Figure 6.13. The histograms confirm that Thrun's ISM is most likely to underestimate occupancy probabilities and create false negatives, as is evident from the high peak at -0.5. This occurs when map cells are marked empty that are not visible in the ground truth.



(a) (b) (c)

Figure 6.13: Map error histograms for the 3D Tsukuba dataset [21], (a) using Thrun's ISM, (b) Andert's ISM, and (c) using PRISM.

Contrary to this, Andert's ISM has significant frequencies of errors around 0, near 0.5, and at 1.0. This confirms that it causes false positives by overestimating occupancy probabilities. PRISM causes errors similar to those in the 2D tests, with errors almost exclusively between 0.0 and 0.5.

These histograms suggest again that the more conservative mapping done using PRISM with a natural maximum influence distance can lead to less problematic errors for navigation. It seems to cause relatively few false negatives, but the question remains of whether it creates useful maps. By only emptying a short distance in front of the sensor, it avoids incorrectly altering the map but the histogram suggests that its limited range leads to overestimations for empty regions in the map. The further distances that we expect to find in the KITTI dataset may improve this range, which should result in more useful maps.

Another noteworthy aspect of the maps, especially of the ones created using Thrun's ISM and Andert's ISM, is their tendency to mark map cells at the thresholds of empty and occupied. It is difficult to see from the 2D maps, but it is an effect of the density of the measurements. This density means that many concurring measurements are added to the map, and their cumulative effect makes the occupancy probabilities reach either the empty or occupied threshold inherent to Octomap. This causes the peaks in the map error histograms, because the ground truth map is ideal and therefore discrete.

6.3.3 Real World 3D Dataset

In order to evaluate the performance of PRISM on real world data we make use of the KITTI dataset [12]. This dataset offers rectified stereo images as well as measurements from an accurate LIDAR. Data was collected in a cluttered, but mostly static urban environment that provides good conditions for occupancy grid mapping.

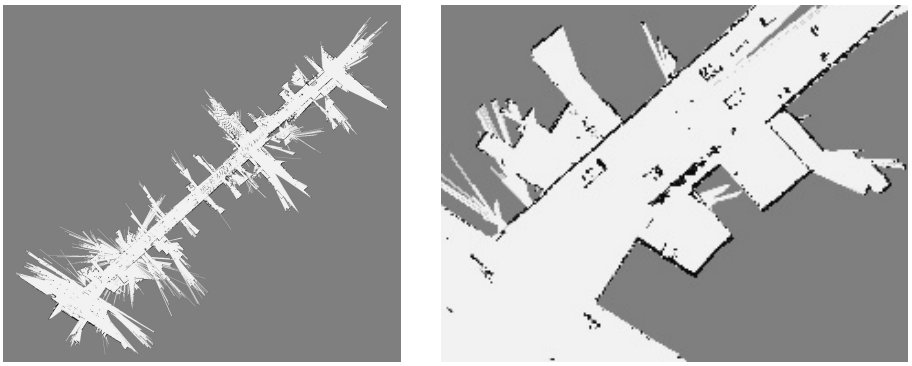
The vehicle also includes a sophisticated OXTS inertial measurement unit (IMU) system with an accurate GPS, which offers position estimates with an accuracy of between 20 cm and 30 cm, and 0.1° accuracy for attitude.

The stereo images are processed using the same dense stereo correspondence algorithms we used in

previous tests. There is no ground truth available for the range information or map, and therefore data from the LIDAR is used for comparison similarly as with the characterisation in Section 3.3.3.1. This laser system has an accuracy of 2 cm over its range, and its measurements are considered correct.

6.3.3.1 LIDAR Ideal Model

As a demonstration of the dataset, two 2D slices of maps drawn using an ideal model of the LIDAR's data are shown in Figure 6.14. The environment consists of a long street with walls on the sides, and a number of side streets. The vehicle turns into this street and drives down it.



(a) (b)

Figure 6.14: (a) Map drawn using ideal sensor model of the entire KITTI LIDAR dataset, (b) a close-up of the map created from the first 50 frames.

The measurements that cause the isolated white lines are mostly because of the laser pointing over visible obstacles or through windows. Some of these measurements, however, are due to reflective surfaces such as glass or water. There are only a few of these incorrect measurements, however, and the overall map can still be considered very accurate.

The few cells in the road marked occupied are due to the 2D slice being taken at the same height throughout. Some sections of the road are inclined, and hence are occupied on the height of the 2D slice. The horizontal field of view of the LIDAR is 360° but that of the stereo vision sensor is much narrower, which means that the visible part of the environment is much less for the stereo vision sensor.

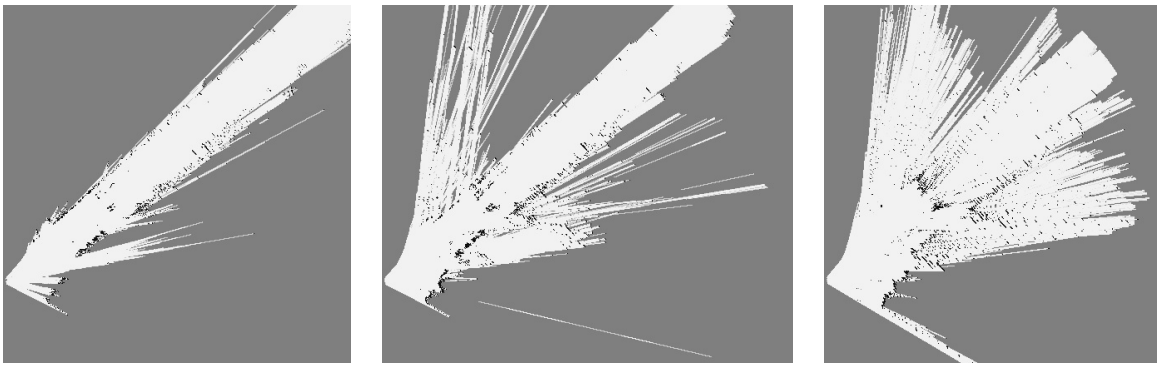
For testing we use the map resulting from the first 50 frames, as shown in Figure 6.14(b).

6.3.3.2 Stereo Vision

The stereo images from the KITTI dataset are used to create maps using a variety of different sensor models. An example of a stereo pair from the dataset is shown in Figure 6.15, displaying the street being mapped.

Figure 6.15: Sample stereo image pair from the KITTI dataset.

The three stereo matching algorithms used are block matching, semiglobal block matching and variational matching, each shown with maps created using the ideal sensor model in Figure 6.16.



(a) (b) (c)

Figure 6.16: The KITTI set mapped (a) using block matching, (b) semiglobal block matching, and (c) variational matching.

It suggests that the block matching method contains fewer errors to the sides of the road, most likely due to its sparser nature, and the variational matching technique appears to contain the most incorrect matches. Note that in these maps some measurement beams may appear to not be connected to the sensor location, since the 2D slice is not made exactly at the height of the sensor.

The maps created using each inverse sensor model follow, where each sensor model has been characterised to the data from each stereo correspondence algorithm. To optimise the maps drawn with the inverse sensor models other than PRISM, their maximum influence distances are limited to 50 metres. Although it is common to limit the influence range [47], not much information is available in the literature regarding the distance at which to do this.

Block Matching

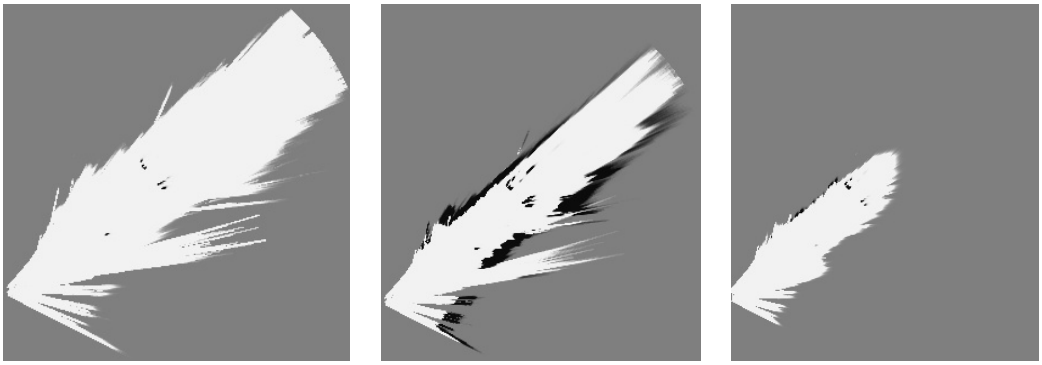
Results from the block matching algorithm are shown in Figure 6.17. Although the map created using Thrun's ISM is mostly empty, the two parked cars and a small section of wall are marked. Many of the obstacles are not marked as occupied, however, which may be a problem for navigation. Andert's ISM calculates higher probabilities of occupancy, and therefore more parts of that map are marked as occupied. This is a better option for navigation, since most of the obstacles to be avoided are indicated.

The area of the map created with PRISM marked unoccupied is significantly smaller than the others due to its limited influence distance. Although the wall to the left and the parked cars are clearly shown, the wall to the right is not visible. From visual inspection this map looks the most similar to the ground truth, but the map error histograms will provide a quantitative comparison.

Semiglobal Block Matching

Figure 6.18, showing the results of mapping with measurements from the semiglobal block matching algorithm, suggests that the algorithm produced more incorrect matches – possibly because it is dense.

The maps drawn using both Thrun's and Andert's ISMs are very similar to the block matching maps, but contain more empty sections to the sides of the road. Both of the maps, but particularly the one using Thrun's ISM, have large incorrectly open areas. The map drawn with PRISM in Figure 6.18(d) contains fewer cells marked unoccupied incorrectly to the sides, and the two parked cars in the road, as



(a) (b) (c)

Figure 6.17: Stereo data from the KITTI dataset with the block matching algorithm is used to draw a number of maps. (a) is created using Thrun's ISM, (b) Andert's ISM, and (c) PRISM.

well as the left wall, are indicated.

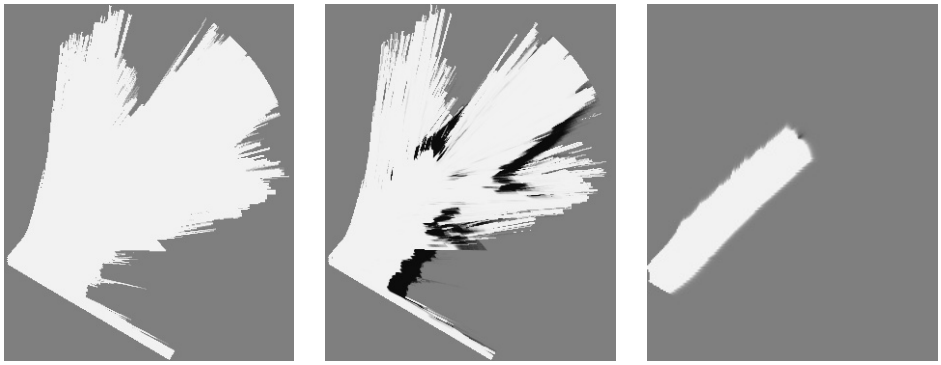
(a) (b) (c)

Figure 6.18: Stereo vision data from the KITTI dataset with semiglobal block matching is used to draw maps. (a) is created using Thrun's ISM, (b) Andert's ISM, and (c) PRISM.

Variational Matching

The measurements from the variational matching algorithm have been shown to be complete but relatively inaccurate. Because the data is so dense, however, many points are usually measured from each obstacle.

The maps shown in Figure 6.19(a) and Figure 6.19(b) would be impractical for navigation, since Thrun's ISM empties large parts of the map and Andert's ISM makes the left side of the road seem like the least obstructed route. The map created using PRISM, in contrast, is more conservative and only contains a small opened area in the middle of the road. It did not mark the obstacles in the road, however, and neither of the walls on the sides of the road. These false negatives could reduce the usefulness of the result for navigation.



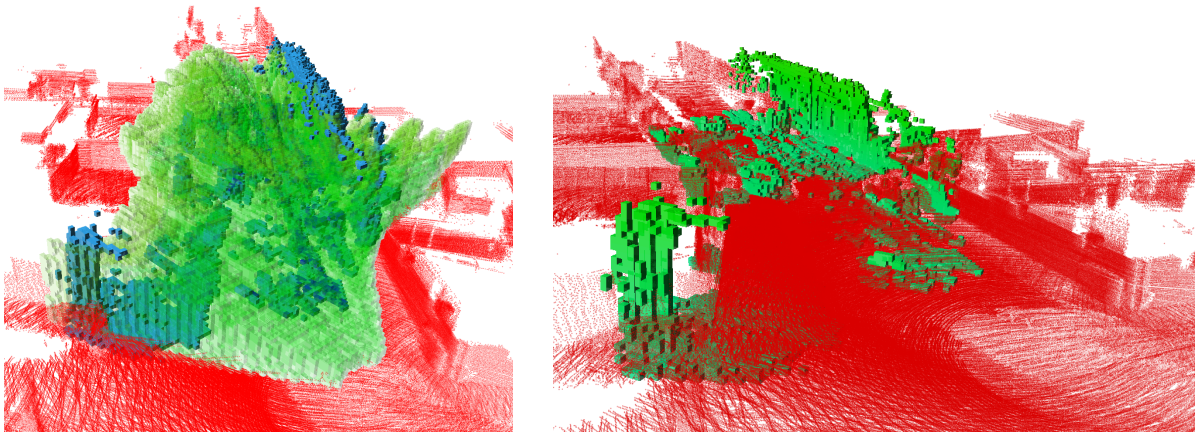
(a) (b) (c)

Figure 6.19: Stereo measurements from the KITTI dataset with variational matching is used to draw a number of maps. (a) is created using Thrun's ISM, (b) Andert's ISM, and (c) PRISM.

6.3.3.3 Full 3D Map Visualisation

Although the 2D slices of 3D maps used up to now are more convenient to analyse, they cannot represent the entire map. Therefore, some visualisations of the full 3D maps are shown here as illustration of the full map. To qualitatively evaluate the accuracy of maps, the LIDAR data is superimposed on the map created using the stereo vision data.

The map shown in Figure 6.20 is of the block matching measurements mapped using PRISM. Since the map was shown to not have many occupied cells, the empty sections are highlighted here in green, indicating that the map is not emptied much outside the empty road area. Note that the part near the middle right of Figure 6.20(a), where the map seems to extend past the LIDAR data, is of a small wall, over which the stereo sensor could measure but the LIDAR could not.



(a) (b)

Figure 6.20: 3D map created of KITTI dataset using block matching, showing the LIDAR data in red. (a) shows the empty areas with green and the occupied with blue, (b) shows just the map cells marked occupied in green.

The map in Figure 6.20(b) is drawn using only the map cells marked occupied by PRISM, in shades

of green. It shows that the walls to the left bottom and towards the far right were both marked in the calculated map. The road surface (as well as some vehicles that are difficult to see in this visualisation) seems to also be clearly captured in the map.

6.3.3.4 Map Error Histograms

To evaluate the accuracy of the calculated maps, the histograms of the differences between the maps using each model and the LIDAR map are shown in Figure 6.21. These histograms contain the error data for all three stereo correspondence algorithms used.

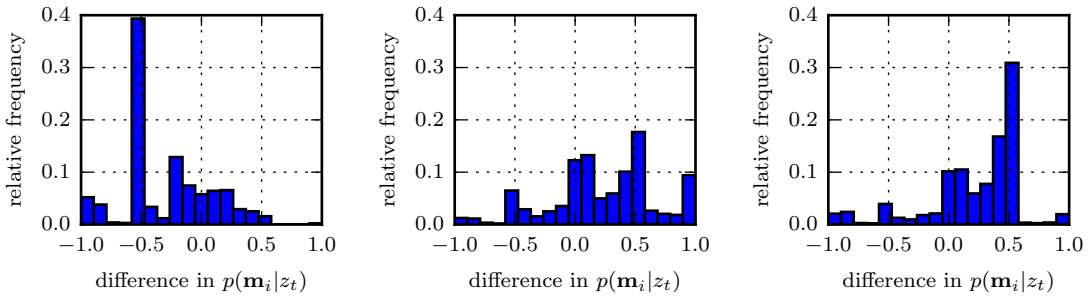


Figure 6.21: Histograms for the errors made in the mapping of the 3D KITTI dataset. (a) uses Thrun's ISM, (b) uses Andert's ISM, and (c) uses PRISM.

The histograms are fairly similar to the 2D simulation. Thrun's ISM seems to be prone to false negatives, Andert's ISM is more prone to false positives and PRISM has mostly false positives smaller than 0.5. This indicates again that Thrun's ISM empties unknown and occupied cells, and that the other two mark empty cells as occupied or unknown.

There are a number of advantages to mapping with PRISM that can be read from these histograms. False positives under 0.5 indicate situations where the LIDAR emptied the map, but the stereo data was not considered accurate up to the measurement range. This is a symptom of the maximum influence distance that is inherent to PRISM.

Although Andert's ISM is shown to cause similar differences, its tendency to mark empty cells as occupied (difference value of 1.0) could create problems for navigating in tight spaces. It is important to note that the differences between maps from different sensor models are smaller for the KITTI dataset than the Tsukuba set. This is because the sensor and map parameters led to a further maximum influence distance, and arguably therefore more useful maps.

Chapter 7

Conclusions

The main focus of this study was to create 3D maps from stereo vision data in a reliable way. To accomplish this, the errors made by stereo vision sensors needed to be analysed and modelled. The usefulness of dense stereo for autonomous mapping in general was also evaluated.

In this chapter we draw some conclusions and present suggestions for further work.

We started with a literature study, discussing some popular stereo correspondence algorithms from literature to get an overview of the field. We chose three representative techniques: block matching, semiglobal block matching and variational matching. After investigating some mapping concepts, the occupancy grid mapping technique was picked for its ability to store probabilities, elegant updating, and because it can distinguish between empty and unknown regions. Some available sensor models that can be used to incorporate new measurements into an occupancy grid were also discussed, after which the inverse sensor model was chosen for its simplicity in incorporating updates.

The geometry used in stereo vision for projecting measurements from the stereo images to real world coordinates was discussed. The next step was to characterise the errors made by stereo vision sensors so the errors of a stereo vision sensor could be modelled. The Gaussian mixture model was chosen to represent the error distributions for its universality in representing different distributions. By fitting a GMM to a training set of error values, we were able to statistically model the errors made by a stereo vision sensor when processed using a particular stereo correspondence algorithm.

When the errors of the stereo correspondence algorithms applied to the synthetic dataset from Tsukuba were characterised, we found that both the block matching based stereo correspondence algorithms are dominated by a single Gaussian component with near zero mean. The variational technique, on the other hand, benefits from a mixture model with multiple components, due to its more complex error profile.

We tested the characterisation on data from the KITTI dataset too, and found that the two block matching based correspondence algorithms again had similar results. The error distribution from the variational matching technique suggested that it makes significantly larger errors.

Next, we investigated the occupancy grid mapping technique, with specific focus on its update equation that incorporates new measurements into the map. This included the introduction of the sensor model used to integrate new measurements into a map. We discussed some assumptions made in the mapping technique as well as some specifically relevant to our mapping implementation. An important assumption we made at this stage is that the measurements can be modelled as 1-dimensional rays, which is only valid as long as the measurement angle is small and accurately known, but greatly simplifies the integration of measurements.

After this we looked at some existing derivations from the literature for inverse sensor models as used with the occupancy grid mapping technique. It was clear that a derivation that is based on physically meaningful and clear assumptions is not available. The available models also all seemed to be limited by an assumption of a unimodal Gaussian error distribution.

Therefore, we provided a principled derivation for the inverse sensor model (PRISM) that allows us to calculate probabilities of occupancy based on physically meaningful assumptions about the environment and measurements. Varying the different parameters that describe the model indicated that it creates sensible models over a variety of measurement distances, cell sizes and measurement noise levels. This also indicated a maximum influence distance inherent in this model for stereo vision sensors, which implied that these sensors naturally have a limited distance to which they can measure objects. They are not capable of providing any information about objects further than that distance, and hence the sensor model only allows the map to be affected up to a certain distance from the sensor. This maximum distance was found to depend on the focal length, baseline and noise variance of the sensor, as well as the map cell size.

To illustrate a further use for PRISM, we made a numerical comparison between it and some popular models from the literature. This indicated that some models offer good approximations at closer ranges and increasing differences at further distances, while the model used in GPSlam [48] has a converging error. We attributed this behaviour to its models converging at far ranges, instead of emptying the map nearly unbounded with distant measurements like the other models.

Next, we tested PRISM by creating maps from a number of 2D and 3D datasets and comparing the results to those of other sensor models. We found that it can accurately model a simple scenario, reliably locating near obstacles and only emptying the map in the directions of further measurements as far as the parameters of the sensor allowed. Some more complex scenarios, including the 3-dimensional real world KITTI dataset, were also well mapped with data from the block matching stereo algorithm. Even with the more complete semiglobal block matching algorithm our inverse sensor model was found to create maps that appeared to represent the ground truth fairly well.

However, we found that measurements from some of the stereo correspondence algorithms (especially on the synthetic images) are simply too inaccurate to yield useful maps. In the case of the variational matching algorithm the sensor characterisation indicated large errors and the sensor model only emptied a small area around the sensor at each time step.

We created a histogram of the mapping errors made by each sensor model, with the intention of capturing what types of errors each model makes. We learnt that the Thrun's ISM is most likely to create false negatives, Andert's ISM is slightly inclined to false positives, and that PRISM largely overestimated occupancy probabilities by marking empty cells as unknown. This suggests that it affecting the map to a maximum influence distance yields acceptable results, since when the other models alter the map further away it leads to more problematic mapping errors.

From our tests in mapping dense stereo measurements with various sensor models, it would appear that stereo vision is a viable sensor for dense mapping. We were capable of creating seemingly useful maps for navigation using measurements from stereo vision sensors, provided that the measurement error distribution was properly characterised and captured by a principled inverse sensor model. It would seem, however, that for mapping the number of measurements from dense stereo may be superfluous. The large number of measurements per time step caused many of the calculated occupancy probabilities to reach upper or lower bounds within a single time step. Since the density of these measurements seems to be a trade-off with accuracy, it seems that sparser data might be more useful for mapping.

7.1 Future Work

Although we showed that our inverse sensor model can be used for mapping, we can improve its usefulness by optimising the implementation. We showed that it is very computationally expensive, and therefore finding an adequate approximation would improve its applicability. Since it involves a number of integrals that need to be solved numerically with a high degree of accuracy, in its current implementation it is mostly useful from a theoretical standpoint due to the correctness of its calculated probabilities. If we are able to approximate, simplify, or pre-calculate the integrals, the general applicability of the model should improve.

One of the assumptions we base our model on is that any obstacle present in a map cell is guaranteed to create a measurement when a measurement ray intersects that cell. Relaxing this assumption may introduce a probability that each map cell contains an unmeasured obstacle. Another assumption that could be addressed is the independence assumption, which is inherent to the standard occupancy grid mapping technique. One way to avoid this assumption may be to incorporate neighbouring cells under a first-order Markov assumption.

Although we derived the sensor model here specifically for a stereo vision sensor, it can be extended to support nearly any range sensor. We expect this would require a new distribution for the measurement, as well as a re-evaluation of the assumptions made regarding measurements. For instance, the angle certainty of sonar measurements could be too low for the 1D measurement beam assumption, which would change the way the derivation is done.

An alternative solution for creating a map from stereo vision to be investigated is to use sparser information to create a differently structured map, possibly using graphs. This map could consist of only specific known obstacles and an extracted road profile, instead of a dense map structure that attempts to describe each map cell individually. Although this would create a less complete map of the environment, it may prove to be sufficient for safe navigation.

Implementing a navigation algorithm was not in the scope of this thesis, but would provide significant insight into the mapping ability of our model. If path planning and obstacle avoidance algorithms are applied to maps created using PRISM, another aspect of the model will be tested.

Nevertheless, when an accurate correspondence algorithm is used, stereo vision has been shown to be capable of densely mapping an environment. Especially when our new inverse sensor model is used to calculate the incremental probabilities of occupancy, useful 3D maps can be created from stereo vision measurements.

Bibliography

[1] S. Thrun, D. Hahnel, D. Ferguson, M. Montemerlo, R. Triebel, W. Burgard, C. Baker, Z. Omohundro, S. Thayer, and W. Whittaker, "A system for volumetric robotic mapping of abandoned mines," IEEE International Conference on Robotics and Automation, vol. 3, 2003.

[2] P. Debanne, J.-V. Herve, and P. Cohen, "Global self-localization of a robot in underground mines," 1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation, vol. 5, 1997.

[3] S. B. Goldberg, M. W. Maimone, and L. Matthies, "Stereo vision and rover navigation software for planetary exploration," in Proceedings of IEEE Aerospace Conference, vol. 5, 2002, pp. 2025–2036.

[4] L. Matthies, M. Maimone, A. Johnson, Y. Cheng, R. Willson, C. Villalpando, S. Goldberg, A. Huertas, A. Stein, and A. Angelova, "Computer vision on Mars," International Journal of Computer Vision, vol. 75, pp. 67–92, 2007.

[5] R. H. Taylor, "A perspective on medical robotics," Proceedings of the IEEE, vol. 94, no. 9, pp. 1652–1664, 2006.

[6] L. Gao, L. Lin, G. Yan, and Rongrong, "Advance in medical robot," Chinese Journal of Medical Instrumentation, vol. 21, pp. 341–344, 1997.

[7] A. Elfes, "Sonar-based real-world mapping and navigation," IEEE Journal on Robotics and Automation, vol. 3, 1987.

[8] S. Se, D. Lowe, and J. Little, "Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks," pp. 735–758, 2002.

[9] J. Steckel and H. Peremans, "BatSLAM: simultaneous localization and mapping using biomimetic sonar," PLoS ONE, vol. 8, no. 1, p. e54076, 2013.

[10] J. Mascaro, M. Detto, G. P. Asner, and H. C. Muller-Landau, "Evaluating uncertainty in mapping forest carbon with airborne LiDAR," Remote Sensing of Environment, vol. 115, pp. 3770–3774, 2011.

[11] M. Ruhnke, R. Kümmerle, G. Grisetti, and W. Burgard, "Highly accurate maximum likelihood laser mapping by jointly optimizing laser points and robot poses," in Proceedings of the IEEE International Conference on Robotics and Automation, 2011, pp. 2812–2817.

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets Robotics : The KITTI Dataset," International Journal of Robotics Research, no. October, pp. 1–6, 2011.

[13] J.-S. Gutmann, M. Fukuchi, and M. Fujita, "3D perception and environment map generation for humanoid robot navigation," The International Journal of Robotics Research, vol. 27, no. 10, pp. 1117–1134, Oct. 2008.

[14] W. Brink, "Stereo vision for simultaneous localization and mapping," Master's thesis, Stellenbosch University, 2012.

[15] M. Agrawal, K. Konolige, and R. Bolles, "Localization and mapping for autonomous navigation in outdoor terrains: a stereo vision approach," in 2007 IEEE Workshop on Applications of Computer Vision (WAC V '07). IEEE, Feb. 2007, p. 7.

[16] W. Zhang and J. Kosecka, "Image based localization in urban environments," in Third International Symposium on 3D Data Processing, Visualization, and Transmission, 2006, pp. 33–40.

[17] S. Thrun, W. Burgard, and D. Fox, Probabilistic Robotics, ser. Intelligent robotics and autonomous agents, S. Thrun, W. Burgard, and D. Fox, Eds. MIT Press, 2005, vol. 45, no. 3.

[18] I. Dryanovski, W. Morris, and J. Xiao, "Multi-volume occupancy grids: an efficient probabilistic 3D mapping model for micro aerial vehicles," International Conference on Intelligent Robots and Systems, no. 1, pp. 1553–1559, 2010.

[19] D. Joubert, "Adaptive occupancy grid mapping with measurement and pose uncertainty," Master's thesis, Stellenbosch University, 2012. [Online]. Available: http://scholar.sun.ac.za/handle/10019.1/71911

[20] D. Marr and T. Poggio, "A computational theory of human stereo vision." Proceedings of the Royal Society of London Series B Containing Papers of a Biological Character Royal Society Great Britain, vol. 204, no. 1156, pp. 301–328, 1979.

[21] University of Tsukuba, "Download New Tsukuba Stereo Dataset." [Online]. Available: http://www.cvlab.cs.tsukuba.ac.jp/dataset/tsukubastereo.php

[22] D. Scharstein, R. Szeliski, and R. Zabih, "A taxonomy and evaluation of dense two-frame stereo correspondence algorithms," Proceedings of Workshop on Stereo and Multi-Baseline Vision, no. 1, pp. 131–140, 2002.

[23] M. Z. Brown, D. Burschka, and G. D. Hager, "Advances in computational stereo," Pattern Analysis and Machine Intelligence, IEEE Transactions on, vol. 25, no. 8, pp. 993–1008, 2003.

[24] K. Konolige, "Small Vision Systems: Hardware and Implementation," in Robotics Research, Y. Shirai and S. Hirose, Eds. Springer London, 1998, pp. 203–212.

[25] H. K. Nishihara, "PRISM: a practical real-time imaging stereo matcher," Cambridge, MA, USA, Tech. Rep., 1984.

[26] D. Gennery, "Modelling the environment of an exploring vehicle by means of stereo vision," Ph.D. dissertation, Stanford, CA, USA, 1980.

[27] H. Hirschmüller, "Stereo processing by semiglobal matching and mutual information," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no. 2, pp. 328–341, 2008.

[28] S. Birchfield and C. Tomasi, "Depth discontinuities by pixel-to-pixel stereo," International Journal of Computer Vision, vol. 35, no. 3, pp. 269–293, 1999.

[29] Y. Ohta and T. Kanade, "Stereo by intra-and inter-scanline search using dynamic programming," Transations on Pattern Analysis and Machine Intelligence, vol. PAMI-7, no. 2, pp. 139–154, 1985.

[30] J. Sun, N. Zheng, and H. Shum, "Stereo matching using belief propagation," Transations on Pattern Analysis and Machine Intelligence, vol. 25, no. 7, pp. 787–800, 2003. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1206509

[31] Q. Yang, L. Wang, and N. Ahuja, "A constant-space belief propagation algorithm for stereo matching," Proceedings of Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1458–1465, Jun. 2010.

[32] S. Kosov, "Multiview 3D reconstruction with variational method," Ph.D. dissertation, Department of Computer Science, Saarland University, 2008.

[33] O. P. Agrawal, "Formulation of Euler–Lagrange equations for fractional variational problems," Journal of Mathematical Analysis and Applications, vol. 272, no. 1, pp. 368–379, 2002.

[34] G. Bradski and A. Kaehler, Learning OpenCV: Computer Vision with the OpenCV Library, 2008, vol. 1.

[35] A. Howard, D. Wolf, and G. Sukhatme, "Towards 3D mapping in large urban environments," International Conference on Intelligent Robots and Systems, vol. 1, pp. 419–424, 2004.

[36] J. Pan, S. Chitta, and D. Manocha, "Probabilistic collision detection between noisy point clouds using robust classification," in International Symposium of Robotics Research, 2011.

[37] D. Kortenkamp and T. Weymouth, "Topological mapping for mobile robots using a combination of sonar and vision sensing," in Proceedings of the Twelfth National Conference on Artificial Intelligence, ser. AAAI'94, vol. 2, no. 2, 1994, pp. 979–984.

[38] B. Kuipers and Y.-T. Byun, "A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations," Robotics and Autonomous Systems, vol. 8, no. 1-2, pp. 47–63, 1991.

[39] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," Computer, vol. 22, no. 6, pp. 46—-57, 1989.

[40] S. Thrun, "Learning occupancy grid maps with forward sensor models," Autonomous robots, pp. 111–127, 2003.

[41] S. T. O'Callaghan and F. T. Ramos, "Gaussian process occupancy maps," The International Journal of Robotics Research, vol. 31, no. 1, pp. 42–62, Jan. 2012.

[42] E. Einhorn, C. Schroter, and H. Gross, "Finding the adequate resolution for grid mapping-cell sizes locally adapting on-the-fly," International Conference on Robotics and Automation, pp. 1843–1848, 2011.

[43] D. A. Reynolds, "Gaussian mixture models," Encyclopedia of Biometric Recognition, vol. 31, pp. 1047–64, 2008.

[44] K. Pathak, A. Birk, J. Poppinga, and S. Schwertfeger, "3D forward sensor modeling and application to occupancy grid based sensor fusion," in IEEE International Conference on Intelligent Robots and Systems, 2007, pp. 2059–2064.

[45] O. J. Woodford and G. Vogiatzis, "A generative model for online depth fusion," in European Conference on Computer Vision. Springer, 2012, pp. 144–157.

[46] A. Elfes, "Occupancy grids: a stochastic spatial representation for active robot perception," Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence, pp. 60—-70, 1990.

[47] F. Andert, "Drawing stereo disparity images into occupancy grids: measurement model and fast implementation," 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5191–5197, Oct. 2009.

[48] K. Pirker, M. Rüther, G. Schweighofer, and H. Bischof, "GPSlam: marrying sparse geometric and dense probabilistic visual mapping," Proceedings of the British Machine Vision Conference 2011, pp. 115.1–115.12, 2011.

[49] K. Wurm, "OctoMap: a probabilistic, flexible, and compact 3D map representation for robotic systems," Proceedings of the ICRA 2010 workshop on best practice in 3D perception and modeling for mobile manipulation, vol. 34, no. 3, pp. 189–206, 2010.

[50] K. Schauwecker and A. Zell, "Robust and efficient volumetric occupancy mapping with an application to stereo vision," International Conference on Robotics and Automation, 2014.

[51] M. Perrollaz, J.-D. Yoder, A. Spalanzani, and C. Laugier, "Using the disparity space to compute occupancy grids from stereo-vision," International Conference on Intelligent Robots and Systems, pp. 2721–2726, Oct. 2010.

[52] Y. Li and Y. Ruichek, "Building variable resolution occupancy grid map from stereoscopic system - A quadtree based approach," Intelligent Vehicles Symposium, no. IV, pp. 744–749, Jun. 2013.

[53] A. Souza and R. Maia, "Occupancy-elevation grid mapping from stereo vision," Robotics Symposium and Competition, 2013.

[54] H. Ghazouani, M. Tagina, and R. Zapata, "Robot navigation map building using stereo vision based 3D occupancy grid," Journal of Artificial Intelligence: Theory and Application, vol. 1, no. 3, pp. 63–72, 2009.

[55] Y. Li, "Stereo vision and Lidar based dynamic occupancy grid mapping: application to scenes analysis for intelligent vehicles," 2013.

[56] L. Heng, D. Honegger, G. H. Lee, L. Meier, P. Tanskanen, F. Fraundorfer, and M. Pollefeys, "Autonomous visual mapping and exploration with a micro aerial vehicle," Journal of Field Robotics, vol. 31, no. 4, pp. 654–675, 2014.

[57] L. Matthies and S. Shafer, "Error modeling in stereo navigation," IEEE Journal on Robotics and Automation, vol. 3, no. 3, pp. 239—-248, 1987.

[58] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision. Cambridge University Press, 2003.

[59] R. A. Horn and C. R. Johnson, Matrix Analysis, 1985, vol. 169.

[60] C. Chang, S. Chatterjee, and P. Kube, "On an analysis of static occlusion in stereo vision," in Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition Computer Vision and Pattern Recognition, 1991, pp. 722–723.

[61] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in International Symposium on Information Theory, 1973, pp. 267–281.

[62] D. Scharstein, "Middlebury Stereo Datasets." [Online]. Available: http://vision.middlebury.edu/stereo/data/

[63] D. Joubert, W. Brink, and B. Herbst, "Pose uncertainty in occupancy grids through Monte Carlo integration," in International Conference on Advanced Robotics, Aug. 2013, pp. 1–6.

[64] P. Z. Peebles, Probability, random variables, and random signal principles. McGraw Hill, 1987.

[65] A. Leon-Garcia, "Probability and random processes for electrical engineering. Addisson-Wesley," Reading, 1989.

[66] W. Feller, An Introduction to Probability Theory and Its Applications, Vol. 2. Wiley, 1971.

[67] R. S. Murray and J. Liu, "Mathematical handbook of formulas and tables," Spiegel-Schaums Outline Series, 1968.